AD-A218 799

# NAVAL POSTGRADUATE SCHOOL
## Monterey, California

# THESIS



ORBITAL MECHANICS
A LEARNING TOOL ON THE MAIN FRAME

by

Anthony A. Vraa

September 1989

Thesis Advisor E.A. Milne

Unclassified

security classification of this page

**REPORT DOCUMENTATION PAGE**

| | |
|---|---|
| 1a Report Security Classification Unclassified | 1b Restrictive Markings |
| 2a Security Classification Authority | 3 Distribution Availability of Report<br>Approved for public release: distribution is unlimited. |
| 2b Declassification Downgrading Schedule | |
| 4 Performing Organization Report Number(s) | 5 Monitoring Organization Report Number(s) |
| 6a Name of Performing Organization<br>Naval Postgraduate School — 6b Office Symbol (if applicable) 366 | 7a Name of Monitoring Organization<br>Naval Postgraduate School |
| 6c Address (city, state, and ZIP code)<br>Monterey, CA 93943-5000 | 7b Address (city, state, and ZIP code)<br>Monterey, CA 93943-5000 |
| 8a Name of Funding Sponsoring Organization — 8b Office Symbol (if applicable) | 9 Procurement Instrument Identification Number |
| 8c Address (city, state, and ZIP code) | 10 Source of Funding Numbers<br>Program Element No — Project No — Task No — Work Unit Accession No |

11 Title (include security classification) ORBITAL MECHANICS A LEARNING TOOL ON THE MAIN FRAME

12 Personal Author(s) Anthony A. Vraa

| 13a Type of Report | 13b Time Covered | 14 Date of Report (year, month, day) | 15 Page Count |
|---|---|---|---|
| Master's Thesis | From To | September 1989 | 99 |

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

| 17 Cosati Codes | | | 18 Subject Terms (continue on reverse if necessary and identify by block number) |
|---|---|---|---|
| Field | Group | Subgroup | orbital mechanics, unperturbed and perturbed orbit, FORTRAN program, DISSPLA graphics. (EG) |

19 Abstract (continue on reverse if necessary and identify by block number)

This thesis consists of an interactive program that enables the student to study the orbital motion of satellites around the earth. The student can investigate the shape of a variety of orbits by varying the initial position and velocity of the satellite, or by supplying select orbital parameters i.e. initial orbital radius, eccentricity, and inclination. Satellite maneuvers can also be studied, like transfer orbits and inclination changes, by command velocity changes at any location in the orbit. Also the effects of the perturbing forces due to the oblateness of the earth, drag for low earth orbits, and gravitational attraction from the sun and moon can be investigated. The orbits are displayed in either the perifocal coordinate system around a model of the earth, or the ground track can be displayed on a map of the world. Orbital data is displayed below the orbital plot. The display is enabled by the use of display integrated software system and plotting language (DISSPLA) subroutines.

| | |
|---|---|
| 20 Distribution Availability of Abstract<br>☒ unclassified unlimited ☐ same as report ☐ DTIC users | 21 Abstract Security Classification<br>Unclassified |
| 22a Name of Responsible Individual<br>E.A. Milne | 22b Telephone (include Area code) (408) 646-2886 — 22c Office Symbol 61Mn |

DD FORM 1473,84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete — security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

# Orbital Mechanics
# A Learning Tool On The Main Frame

by


Anthony A. Vraa
B.S., University of Minnesota, 1981


Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY
(SPACE SYSTEMS OPERATIONS)

from the

NAVAL POSTGRADUATE SCHOOL
September 1989

Author: Anthony A Vraa
Anthony A. Vraa

Approved by: Edmund A. Milne
E.A. Milne, Thesis Advisor

D. A. Danielson
D.A. Danielson, Second Reader

R Panholzer
Rudolph Panholzer, Chairman,
Department of Space Systems

# ABSTRACT

This thesis consists of an interactive program that enables the student to study the orbital motion of satellites around the earth. The student can investigate the shape of a variety of orbits by varying the initial position and velocity of the satellite, or by supplying select orbital parameters i.e. initial orbital radius, eccentricity, and inclination. Satellite maneuvers can also be studied, like transfer orbits and inclination changes, by command velocity changes at any location in the orbit. Also the effects of the perturbing forces due to the oblateness of the earth, drag for low earth orbits, and gravitational attraction from the sun and moon can be investigated. The orbits are displayed in either the perifocal coordinate system around a model of the earth, or the ground track can be displayed on a map of the world. Orbital data is displayed below the orbital plot. The display is enabled by the use of display integrated software system and plotting language (DISSPLA) subroutines.

## TABLE OF CONTENTS

# I. INTRODUCTION

A visual aid for students new to orbital mechanics is required to comprehend fully the dynamics of orbital motion. This program is an interactive time step simulation program that calculates and plots either unperturbed or perturbed elliptical orbits. The program interacts with the student in developing the initial orbit. Also the program enables the student with the ability to change the velocity of the satellite at a specific location in the orbit. This feature will permit the student to investigate the effects of commanded velocity changes as in perigee kicks, apogee kicks and inclination changes. The user can also modify the initial position and velocity of the satellite at the completion of any orbit.

The student is given an opportunity to investigate the effects of perturbing forces on the satellites orbit by choosing to have the program calculate the orbit with or without perturbing forces. The variation of parameters method, as seen in [Ref. 1: pp. 396-407], is used in calculating the perturbing orbit. The perturbing forces taken into consideration are the following:

1. the oblateness of the earth
2. drag for low earth orbits
3. gravitational force of the moon
4. gravitational force of the sun

In order to review fully the operation of the program (included in appendix A) and to uncover any problems or limitations that plagued the programming, the program has been divided up as follows:

1. program design
2. unperturbed orbit
3. perturbed orbit
4. velocity changes
5. graphical plots

The programming approach and equations used in each of the above sections will be examined in there respective chapters. A review of the coordinate systems used and their

transformations between them are included in appendix B. Since all the equations used in the calculation of the orbital elements are from reference 1, they will not be reviewed in each chapter but will be included in appendix C for a quick reference. Equations from other sources will be referenced in their respective chapters.

Examples of perturbed and unperturbed orbital plots for a variety of initial orbital parameters are inc  d in appendix D. Included are plots of low earth orbits, transfer orbits and geosynchronous orbits.

# II. PROGRAM DESIGN

In designing this program an attempt was made to make it not only as user friendly as possible, but also to make the program as simple as possible to understand. To achieve these goals, the program would have to be written in a logical manner, in a computer language that is easy to follow, the program would have to run on terminals readily available to students (at the Naval Postgraduate School (NPS)), and the program would have to be easily used by students with a minimum amount of computer or orbital mechanics knowledge.

FORTRAN was chosen as the programming language since it is a wildly used scientific language and it allows for very structured programming. By programming in a structured format, the program can be expanded in the future with a minimum amount of time required to understand the programming code. FORTRAN also allows for double precision numbers to be used in the calculation of the orbit. This is critical when round off error in single precision could be greater then the actual change that one is trying to model. The equations in the descriptions of the program might not exactly match the equations in the listings because of special programming techniques which must be included in most computer programs to handle such problems as "division by zero".

The display integrated software system and plotting language (DISSPLA) package available on the mainframe computer at NPS was used to enable a variety of graphical displays with a minimum amount of programming. DISSPLA has a set of subroutines that the programmer calls to display data contained in arrays. This requirement forces the program to load arrays with the satellites position in order for it to be plotted. The TEC618 computer terminal and associative plotter was used for ease of gaining hard copy plots of the orbits and the diversity of locations that are available here at NPS. In order to run a program in DISSPLA the user must first define storage space of 1500k and designate temporary disk space, and then call DISSPLA with the program name. This is accomplished with the following commands:

1. DEFINE STORAGE 1500K

2. LCMS
3. TDISK 4 DIS
4. DISSPLA ORBIT

To make the program user friendly, the user is prompted for inputs via the keyboard. The entry is usually a number. A yes or no response can be entered by typing "Y" or a "N". In most cases the program does a check to see if the input is appropriate. In order to make it as easy as possible for the student to get the desired orbit displayed, the program requires only the initial position and velocity of the satellite. The initial position and velocity of the satellite is supplied by the user in one of two ways. The user can input the position and velocity of the satellite, using the perifocal coordinate system (IJK), or the user can let the program place the satellite on the "I" axis of the IJK system at the radius of perigee (RP) distance supplied by the user. This latter choice gives the initial location of the satellite, but to get the velocity the program will prompt the user for one of the following:

1. the actual velocity in the IJK system.
2. the eccentricity (e) of the orbit. In which case the velocity is calculated from the following equations:

   $a = \frac{RP}{1-e}$ = semi-major axis

   $ENR = -\frac{\mu}{2a}$ = energy mass

   Where $\mu$ = MG
   M = mass of earth
   G = Universal gravitational constant

   $v = \sqrt{2(ENR + \frac{\mu}{RP})}$

3. the radius of apogee (RA) The velocity is calculated by first calculating the eccentricity (e) from the following:

   $e = \frac{RA - RP}{RA + RP}$

   With the eccentricity the same equations used above are used to calculate the velocity.

In order to give the velocity a direction the inclination (i) of the orbit is required from the user. The following equations are used to calculate the velocity vector:

$v_I = 0.0$

$v_y = v \cos\theta$

$v_x = v \sin\theta$

The program will check to ensure that the orbital eccentricity is less than 1.0, if it is not then the program will reject the inputs. After the initial input are accepted, the program will do calculations for the six orbital elements required to describe the size, shape and orientation of the orbit, and to pinpoint the position of the satellite along the orbit at a particular time. This classical set of six orbital elements are as follows:

1. a, semi-major axis.
2. e, eccentricity.
3. i, inclination.
4. $\Omega$, longitude of the ascending node.
5. $\omega$, argument of perigee passage.
6. T, time of perigee passage.

The program actually calculates more orbital elements than the six classical elements required to plot the orbit, this is done in an effort to make the program as robust as possible. This will add in the ability to expand the program in the future.

If the satellite is not initially at the perigee point then the satellite [illegible] first be stepped around to the perigee point. The program then enters a loop that calculates the orbit from the perigee point through one complete orbit around the earth and back to the perigee point. The orbit is calculated in steps of 2 times pi divided by an integer, i.e., 2 times pi divided by 50. This step size was used to ensure a smooth orbit for display purposes and also to get within adequate distance to the perigee point or other location for a velocity change. After the loop is completed, the program will offer the user a choice of the following plots to check the orbit:

1. perifocal
2. groundtrack

The program then goes into a loop offering the user the following choices until the user decides to end the program:

1. plot another view of the same orbit.

If the user wishes to plot another view of the same orbit then the user may use this choice to reenter the display portion of the program.

2. plot the next orbit (perturbed or unperturbed).
   To plot the next orbit the satellite is stepped around the complete orbit either with or without perturbing forces effecting the satellite.

3. change the initial conditions.
   The program goes to the beginning of the program and allows the user to change the initial position and velocity of the satellite.

4. change the velocity at a specific location
   Step the satellite around to a specific true anomaly and make a velocity change at that location.

5. clear the previous orbits from the plot.
   Clear the memory of all the previous orbits and only retain the current location and velocity as the initial position and velocity.

Before each new orbit, the orbital elements are recalculated.

There are several common assumptions and constants used throughout the program i.e. all bodies are considered to be spherically symmetric (this allows these bodies to be treated as though their masses are concentrated at their centers (point masses)). other assumptions will be covered in their respective chapters.

## III. UNPERTURBED ORBIT

The subroutines that calculate the unperturbed orbit are the most widely used subroutines in the entire program. These subroutines are called to step the satellite around to the perigee point from the user supplied initial position and velocity, to calculate the next unperturbed orbit, and for any velocity change. No matter which of these sources supply the initial position and velocity the program calculates the unperturbed orbit in the same manner. The only difference is where in the orbit the satellite is initially when these subroutines are called. Before the unperturbed subroutines are called, the orbital elements are calculated.

The unperturbed subroutines are called by a single subroutine 'UNPRET' which has the following basic algorithm:

1. Increment time by the time step size (DT). The time step was chosen as the period divided by fifty to give a smooth plot, but more importantly to ensure that the satellite is within an acceptable distance from a specific location for a velocity change. The angular error caused by the step size can be as much as PI 50 from the desired point for a circular orbit and will increase for more eccentric orbits. This error becomes a factor when the user is making velocity changes, and therefore it will be covered in that chapter in further detail.

2. Calculate the new elements. The calculation of the new elements is the heart of this algorithm. The size, shape and orientation of the orbit remains unchanged. What is required is the position of the satellite along the orbit as a function of time. The problem becomes a matter to solve "the Kepler problem"-predicting the future position and velocity of an orbiting object as a function of some known initial position and velocity and the time of flight [Ref. 1: p. 181]. An algorithm using these principles will follow:

    a. A time step (DT) is added to the time of flight(TF), time of flight is the elapsed time since the satellite passed the perigee point.

        $TF = TF + DT$

    b. The new mean anomaly (MA) is calculated from the new time of flight, and the mean motion (MM).

        $MA = MM \times TF$

    c. With the new mean anomaly the new eccentric anomaly (EA) is calculated. Because the solution to the Kepler problem ($MA = EA - e \times \sin(EA)$) is transcendental, an iterative solution based on the Newton method of root finding is used. The root in question is a solution to the equation ($MA - EA + e \times \sin(EA) = 0$). This algorithm takes the form of [Ref. 1: p. 222]:

        1) $M.A. = E.A. - e \times \sin(E.A.)$

2.

$$EA_{n+1} = EA_n - \frac{(MA - MA_n)}{(1 - e \times \cos(EA_n))}$$

Where this equation is applied initially to $EA_n = MA$ and then reapplied until the difference between MA and $MA_n$ becomes small enough to be ignored.

d. The new true anomaly ($v_o$) is calculated from:

$$v_0 = \frac{\cos^{-1}(e - \cos(EA))}{e \cos(EA) - 1}$$

3. Calculate the new position and velocity. The position and velocity are calculated in the perifocal coordinate system (PQW). The PQW system uses the orbit as its fundamental plane and therefore requires only two coordinate to specify the satellite's position and velocity. The $z_w$ coordinate is by definition always equal to zero. The position of the satellite is calculated as:

$$x_w = r \cos v$$

$$y_w = r \sin v$$

$$z_w = 0$$

The velocity of the satellite is calculated as:

$$v_x = \sqrt{\frac{\mu}{p}}\,(-\sin v_o)$$

$$v_y = \sqrt{\frac{\mu}{p}}\,(e + \cos v_o)$$

$$v_z = 0$$

4. Store position and elements in arrays for plotting. In order for the program to plot the orbit the radius, true anomaly, inclination, and argument of perigee must be stored in arrays. The use of these arrays to plot the orbit will be explained in chapter 6.

5. The process is repeated until the satellite is at the perigee point and the true anomaly is two pi.

The procedure used to calculate the unperturbed orbit leave very little to be modified by a programmer. The only choices that had to be made concerned step size, how to tell the UNPRET subroutine that the perigee point had been reached, and a value of acceptable error for newtons method. For the unperturbed orbit, the step size just had to be small enough to produce a smooth plot of the orbit. Two indicators for perigee were used, one was that the true anomaly was greater than 6.21 radians (two pi equals 6.28 radians) and the time from the previous perigee point will be greater then the period. The two indicators were logically 'and' together to ensure the perigee point was reached.

The disparity between two pi and 6.21 radians is due to the error produced by the satellite not beginning the orbit at exactly the perigee point and the step size used go around the orbit. The acceptable size of error for newtons method was set at 1.0E-10, because for an unperturbed orbit this would be the major contributor to any error in the orbit and the magnitude of this error would be acceptable. However: in a perturbed orbit there are other factors contributing to determining the acceptable error, and these will be discussed in the next chapter.

## IV. PERTURBED ORBIT

The perturbed orbit uses the same basic routines as the unperturbed orbit in stepping the satellite around the earth with one major difference, the perturbing forces produce a time rate of change of the orbital elements that must be applied at each time step. The variation of parameters method is used to determine this influence of the perturbing forces on the orbital elements. The analysis is simplified by using the orbital coordinate system 'RSW', as explained in appendix B. The basic algorithm is as follows [Ref. 1: p. 407]:

1. At $t = t_0$ calculate six orbital elements.
2. Compute the perturbing forces and transform it at $t = t_0$ to the 'RSW' SYSTEM.
3. Compute the time rate-of-change of the elements.
4. Calculate the change of elements for one time step, and add the changes to the old values at each step to get the new elements.
5. From the new values of the orbital elements, calculate a position and velocity.
6. Go to the step 2 and repeat until the final time is reached.

The steps in the algorithm will be explained in the following sections:

### A. ORBITAL ELEMENTS

The standard orbital elements a, e, i, Ω, ω and T (or M) will be used, where

a = semi-major axis

e = eccentricity

i = inclination

Ω = longitude of ascending node

ω = argument of perigee

T = time of perigee passage

($M_e$ = mean anomaly at epoch = $M - n(t - t_e)$). The elements are calculated only at the beginning of the orbit from the initial position and velocity vectors. The elements are then changed continuously throughout the orbit by adding the changes due to the perturbing forces. For the perturbed orbit, the satellite will always begin at the perigee point. This is done so one complete orbit is from perigee point to perigee point.

## B. COMPUTE PERTURBING FORCES

The variation of parameters method requires that the perturbing forces be calculated at each step in the orbit. In order to do this a model of each perturbing force must be developed. The following perturbing forces where used in calculating the total perturbing force effecting the satellite:

1. oblateness of the earth
2. atmospheric drag
3. gravitational attraction of the sun
4. gravitational attraction of the moon

The magnitudes of these forces have an enormous range of values and are dependent on the distance the satellite is from the perturbing body. Figure 1 on page 12 shows a graphical representation of the magnitude of the perturbing forces in a log-log plot of perturbing forces per unit mass [Ref. 2: p. IV-61]. The model of each of these forces follows:

### 1. NON-SPHERICAL EARTH

The earth is not perfectly spherical, but bulges around the equator. The polar and equatorial diameters are 12713.0 Km and 12756.3 Km, respectively. The oblateness results in a perturbing force per unit mass with these components in the 'RSW' coordinate system [Ref. 3: p. 81]:

$$F_r = \frac{(-3\mu J_2 r_e^2)}{2r^4}(1 - 3\sin^2(i)\sin^2(u_0))$$

$$F_s = \frac{(-3\mu J_2 r_e^2)}{r^4}(\sin^2(i)\sin(u_0)\cos(u_0))$$

$$F_w = \frac{(-3\mu J_2 r_e^2)}{r^4}(\sin(i)\cos(i)\sin(u_0))$$

The variable and constants of these equations are defined below:

1. Variables:

   a. $u_0$ = the argument of latitude and is equal to the true anomaly $v_0$ plus the argument of perigee $\omega$.

   $u_0 = v_0 + \omega$

   b. r = the radius from the center of the earth to the satellite.

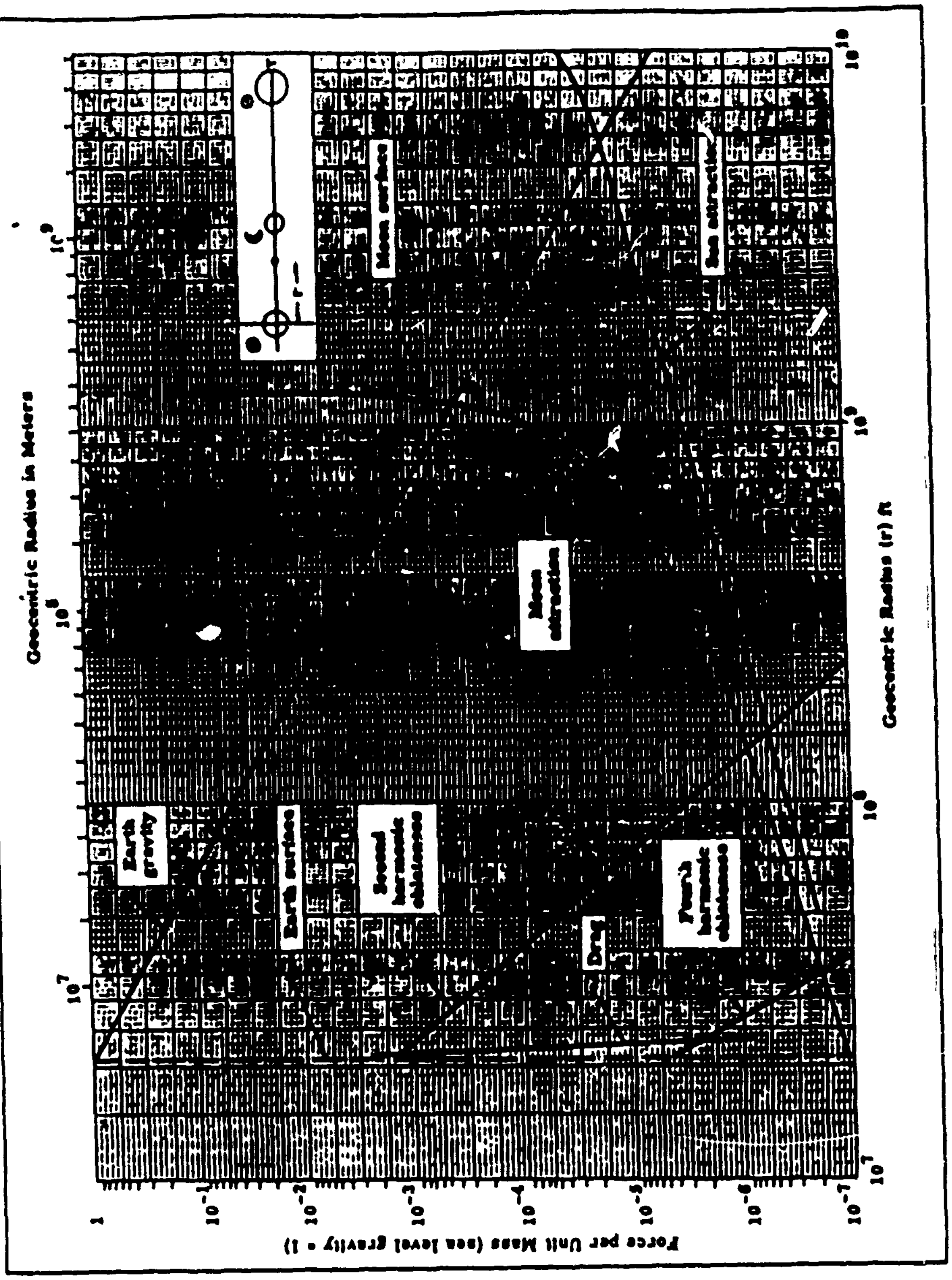



**Figure 1. Comparison of perturbation magnitudes.**

$r = |\vec{r}|$

2. Constants:

   a. $\mu$ = the gravitational parameter of the earth.

   $$\mu = 398601.2 \frac{(km^3)}{s^2}$$

   b. $J_2$ = the second harmonic of oblateness coefficient, determined by experimental observations.

   $J_2 = 1.08263E-3$

   c. $r_e$ = the mean radius of the earth.

   $r_e = 6.37812E3Km$

## 2. ATMOSPHERIC DRAG

The formulation of atmospheric drag equations are plagued with uncertainties of atmospheric fluctuations, frontal areas of orbiting object (if not constant), the drag coefficient, and other parameters. A fairly simple formulation will be given here. Drag, by definition, will be opposite to the velocity of the vehicle relative to the atmosphere. Thus, the perturbing force is

$$\vec{F} = -\left(\frac{1}{2m}\right) \cdot CD \cdot AR \cdot DEN \cdot v \cdot \vec{v}$$

The velocity vector is in the 'IJK' system so the resulting force is also in the 'IJK' system. Therefore a transformation to the 'RSW' system is required.

The variables and constants of this equation are defined below:

1. Variables:

   a. v = speed of vehicle.

   b. CD = the dimensionless drag coefficient. The drag coefficient CD has a value between 1 and 2. It takes a value near 1 when the mean free path of the atmospheric molecules is small compared with the satellite size, and takes a value close to 2 when the mean free path is large compared with the size of the satellite. The drag coefficient will be modeled with CD = 2 when the satellites altitude is greater than 550km and equal to 1 otherwise. [Ref. 4: p. 295]

   c. DEN = atmospheric density at the vehicle's altitude. The density is spherically symmetric, and will be modeled using exponential steps using the parameters in Table 1 on page 14 and the following formula [Ref. 1: pp. 423-424]:

   $$\delta(z) = \delta_0 e^{(-z/H)}.$$

Table 1. ATMOSPHERIC PARAMETERS AND VALUES

| km | $\rho$ | k | h | $\rho(h)$ |
|---|---|---|---|---|
| 0-150 | 1.225E-02 | 4.74E-02 | 0.0 | 1.2225E-02 |
| | | | 150 | 1.0E-03 |
| 150-550 | 1.79846E-01 | 4.3614E-02 | 550 | 3.0E-8 |
| 550 | 1.015484E-07 | 2.2169SE-07 | 1500 | 3.65E-09 |
| | | | 2100 | 1.0E-12 |

2. Constants set to typical values:
   a. m = mass of the satellite, set equal to 100kg.
   b. AR = the sectional area of the vehicle perpendicular to the direction of motion.

### 3. PERT[illegible] DUE TO HEAVENLY BODY

The satellite [illegible] perturbation forces due to the gravitational effects of the sun and the moon. The perturbation force from a perturbing body is the difference between the gravitational force due to the perturbing body at the satellite and the gravitational force the satellite would experience if it were at the center of the earth. From Figure 2 on page 15, the perturbing force per unit mass of the satellite is

$$\bar{f}_p = \mu_p \frac{r_p \bar{i}_p - r\bar{i}_r}{|r_p \bar{i}_p - r\bar{i}_r|^3} - \frac{\mu_p \bar{i}_p}{r_p^2}$$

The variable and constants are defined below:

1. Variables:
   a. $r_p$ = distance from the earth center for the perturbing body
   b. $\bar{i}_p$ = unit vector from the earth to the perturbing body
   c. $r$ = distance from earth center to the satellite
   d. $\bar{i}_r$ = unit vector from the earth to the satellite
2. Constants:
   a. $\mu_p$ = gravitational constant of the perturbing body = $M_p G$

The subscript p is to be replaced by s if the perturbing body is the sun, and by m if the perturbing body is the moon. We will assume that $r << r_p$ then the equation above becomes

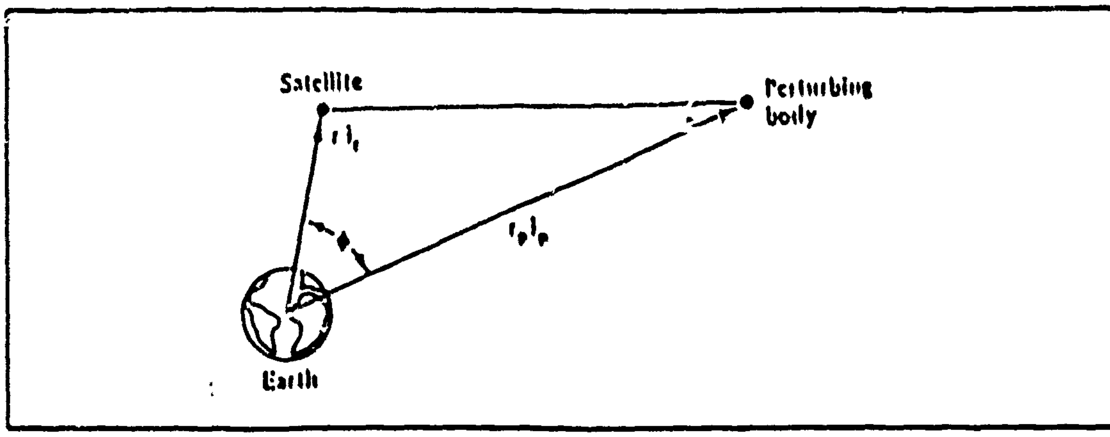



**Figure 2. Perturbation forces.**

$$\vec{F}_p = \left(\frac{\mu_p}{r_p^2}\right)\left(\frac{r}{r_p}\right)\left(3(\overline{i_r i_p})\vec{i}_p - \vec{i}_r\right)$$

The unit vectors $\vec{i}_r$ and $\vec{i}_p$ can be written in terms of the 'IJK' system as:

$$\vec{i}_r = (\cos(\Omega)\cos(u_0) - \sin(\Omega)\cos(i)\sin(u_0))\vec{I} + (\cos(u_0)\sin(\Omega) + \cos(\omega)\cos(i)\sin(u_0))\vec{J} +$$

$$(\sin(i)\sin(u_0))\vec{K}$$

$$\vec{i}_p = (\cos(\Omega_p)\cos(u_{0p}) - \sin(\Omega_p)\cos(i_p)\sin(u_{0p}))\vec{I} +$$

$$(\cos(u_{0p})\sin(\Omega_p) + \cos(\omega_p)\cos(i_p)\sin(u_{0p}))\vec{J} + (\sin(i_p)\sin(u_{0p}))\vec{K}$$

where $\Omega$, i, and $u_0$ are the orbital elements of the satellites and $\Omega_p$, $i_p$, and $u_{0p}$ are the orbital elements of the perturbing body. The formulas above use the 'IJK' system, and as such the resultant forces must be transformed to the 'RSW' system. Models of the sun and moon orbits are required to calculate $\vec{r}_p$ and $\vec{i}_p$. The models used in the program for the sun and moon's orbits follows: [Ref. 3: pp. 73-74]

*a. SUN'S POSITION*

In order to model the suns orbit, a number of simplifications had to be made in the actual parameters of the suns orbit. First the sun will be assumed to be in a circular orbit.This means that the radius (r) to the sun will be constant, and the eccentricity (e) will equal 0.0 instead of its true value of 0.017. The other assumption will

be to place the sun on the 'I' axis of the 'IJK' system at the beginning of the program and have it progress through its orbit as the program runs. These changes will not effect the perturbing force in any noticeable magnitude.

The following variables and constants where used in the program to model the suns orbit after applying the simplifications: [Ref. 3: pp. 75-78]

1. Constants:

   a. Gravitational Constant: $G = 6.67E-11\,\frac{(Nm^2)}{kg^2}$

   b. Sun's Mass: $m_s = 1.99E30Kg$

   c. Sun's Gravitational parameter:

      $\mu_s = 1.32733E20\,\frac{Nm^2}{kg}$

   d. Sun's eccentricity: $e_s = 0.0$

   e. Radius of orbit, assume sun is in circular orbit: $r_s = 1.49E11m$

   f. Sun's inclination: si = 23.45 deg. = 4.09279709d-01 radians

   g. Longitude of ascending node: $\Omega_s = 0.0$

   h. Argument of perigee: $\omega_s = 0.0$

2. Variables:

   a. The true anomaly of the sun's position as a function of the time the satellite has been in orbit:

      $$v_{0_s}(TT) = \frac{2\pi}{356 \times 24 \times 3600}\,TT$$

      Where TT = true time, the time the satellite has been in orbit (sec)

   b. Sun's Position vector: $\bar{r} = r\cos v_{0_s}\bar{P} + r\sin v_{0_s}\bar{Q}$

   c. Unit vector from the earth to the sun: $\bar{i}_s = \frac{\bar{r}_s}{|\bar{r}_s|}$

### *b. MOON'S POSITION*

In modeling the orbit of the moon, similar assumptions where used as with the sun. The moons orbit will be assumed to be circular, actually the eccentricity is equal to 0.055. By placing the moon initially on the 'I' axis of the 'IJK' system along with the sun, the gravitational forces of the two bodies will combine to a maximum. However; since the moons orbital period is only 27.3 days, the moon will not stay in this alignment and the magnitude of the combined forces will vary with time. The inclination of the moons orbit is not constant, but drifts between 18.3 and 28.6 degrees in ten years.

Also the longitude of the ascending node ($\Omega$) oscillates between 13 and -13 degrees. To simplify this the inclination will be chosen as a constant 23.5 degrees and the longitude of the ascending node as 0.0 degrees. For the time period involved in calculating the perturbed orbit, these assumptions will not make any significant difference.

The following variables and constants were used in the program to model the moons orbit, after applying the simplifications:

1. Constants:

   a. Gravitational Constant: $G = 6.67E-11 \frac{(Nm^2)}{kg^2}$

   b. Moon's Mass: $m_{\varepsilon} = 7.35E22kg$

   c. Moon's Gravitational Parameter: $\mu_{\varepsilon} = GM_{\varepsilon} = 4.90E12 \frac{(Nm^2)}{kg}$

   d. Moon's eccentricity: $e_{\varepsilon} = 0.0$

   e. Radius of orbit, assume moon is in circular orbit: $r_{\varepsilon} = 3.844E8km$

   f. Moon's inclination: i = 23.5deg. = 4.10152374E-1 radians

   g. Moon's longitude of ascending node: $\Omega_{\varepsilon} = 0.0$

   h. Moon's argument of perigee: $\omega_{\varepsilon} = 0.0$

   i. Moon's period: $T = 27.3$ days [period]

2. Variables:

   a. The true anomaly of the moon's position as a function of the time the satellite has been in orbit: $v_{\varepsilon}(TT) = \frac{2\pi}{27.3 \times 24 \times 3600} TT$

   b. Moon's position Vector: $\bar{r} = r \cos v_{\varepsilon} \bar{P} + r \sin v_{\varepsilon} \bar{Q}$

   c. Unit vector from earth to moon: $\bar{i}_{\varepsilon} = \frac{\bar{r}_{\varepsilon}}{|\bar{r}_{\varepsilon}|}$

The models of the sun and moons orbit calculates the position vector in the 'PQW' system and therefore the position vector must be transformed to the 'IJK' system.

## C. RATE-OF-CHANGE OF ORBITAL ELEMENTS

The derivations and equations of the rates-of-change of the orbital elements are contained in reference 1 pages 398 to 406. Therefore; only a summary of the actual analytic expressions for the rate-of-change of the parameters in terms of the perturbations will follow:

1. Rate-of-change of the semi-major axis:

$$\frac{da}{dt} = [\frac{2e \sin v}{n'\sqrt{1-e^2}}]F_r - [\frac{2a\sqrt{1-e^2}}{n'r}]F_t$$

Where $n'$ is the mean motion of the satellites orbit.

$$n' = \sqrt{\frac{\mu}{a^3}}$$

2. Rate-of-change of the eccentricity:

$$\frac{de}{dt} = [\frac{\sqrt{1-e^2}\sin v_0}{n'a}]F_r + [\frac{\sqrt{1-e^2}}{n'a^2e}][\frac{a^2(1-e^2)}{r} - r]F_t$$

3. Rate-of-change of the inclination:

$$\frac{di}{dt} = [\frac{r\cos u_0}{n'a^2\sqrt{1-e^2}}]F_w$$

4. Rate-of-change of the longitude of the ascending node:

$$\frac{d\Omega}{dt} = [\frac{r\sin u_0}{n'a^2\sqrt{1-e^2}\,\sin i}]F_w$$

5. Rate-of-change of the argument of perigee:

$$\frac{d\omega}{dt} = (d\frac{\omega}{dt})_r + (d\frac{\omega}{dt})_t + (d\frac{\omega}{dt})_w$$

Where,

$$(\frac{d\omega}{dt})_r = [\frac{-\sqrt{1-e^2}\cos v_0}{n'ae}]F_r$$

$$(\frac{d\omega}{dt})_t = [\frac{r}{eh}][\sin v_0(1 + \frac{1}{1+e\cos v_0})]F_t$$

$$(\frac{d\omega}{dt})_w = [\frac{-r\cot i\sin u_0}{n'a^2\sqrt{1-e^2}}]F_w$$

6. Rate-of-change of the eccentric anomaly:

$$\frac{dEA}{dt} = \frac{1}{\sin(EA)}\frac{[(\sin v_0 + \frac{de}{dt})(1+e\cos v_0) - (\cos v_0 + e)(\frac{de}{dt}\cos v_0 + e\sin v_0)]}{[1+e\cos v_0]^2}$$

7. Rate-of-change of the mean anomaly:

$$\frac{dMA}{dt} = \frac{dEA}{dt} - \frac{de}{dt}\sin(EA) - e \times \cos(EA)\frac{dEA}{dt} - \frac{dn'}{dt}(t - t_0)$$

This equation reduces to the following for circular and ecliptic orbits $(0 < e < 1)$.

$$\frac{dMt}{dt} = \frac{-1}{na}\left[\frac{2r}{a} - \frac{(1-e^2)}{e}\cos v_t\right]F_r - \left[\frac{1-e^2}{nae}\right]\left[1 + \frac{r}{a(1-e^2)}\right]\sin v_t F_t - t\frac{dn'}{dt}$$

Where the Rate-of-change of the mean motion:

$$\frac{dn'}{dt} = \left[\frac{-3n}{2n'a^2}\right]\frac{da}{dt}$$

[ref. 1 p. 396-407]

## D. NEW ORBITAL ELEMENTS

The change of each element is calculated by multiplying the rate-of-change of the element by the time step (DT). The change in the orbital elements are then added to the current values of the elements to give the new orbital elements. With the new elements calculated, the satellite is stepped forward and the new position and velocity are calculated in the same manner as the unperturbed orbit (chapter 3). Also as with the unperturbed orbit, the process is repeated until the satellite is at the perigee point, indicated by the time of flight (TF) equal to the period of the perturbed orbit.

## V. VELOCITY CHANGES

The ability of the student to change the velocity of the satellite at any position in the orbit is a vital element in this program. With velocity changes the student can investigate the effects of varying the satellites velocity as in transfer orbits and inclination changes. In order to simplify the program the unperturbed orbit is used throughout this routine. The velocity change algorithm used in the program follows:

1. Rotate to velocity change location.

   The user is given the choice of changing the velocity of the satellite at the perigee, apogee or at any true anomaly. If the user chooses perigee or apogee as the change locations, the true anomaly is set equal to zero or pi radians respectfully. With the location of the velocity change, the satellite is first stepped around to the desired true anomaly. The stepping is identical with the unperturbed orbit with the exception that the stepping terminates when the true anomaly is greater or equal to the desired true anomaly. With a step size of one fiftieth of the period, the satellite is actually stepped around to a location near the desired location. This variance can be reduced by decreasing the step size but this would increase the computation time. This error will be a major factor in precise calculations of transfer orbits, or any other orbital maneuver where precise velocity changes are required. However: this program is not a tool to calculate precise orbital maneuvers, but rather a learning tool for the student to get a feel for the results of velocity changes in a satellite's orbit.

2. Change the velocity.

   With the satellite at the desired location, the program calculates and displays for the user the satellite's current velocity, escape velocity and circular velocity (the velocity required to circularize the orbit). The program will not allow velocities greater than or equal to the escape velocity. The user is given the option to enter a new velocity in the 'IJK' system or to change the magnitude of the velocity in the orbital plane. If the user chooses to change the velocity in the orbital plane, the program will prompt the user for the magnitude of the velocity change, and multiply this change by a unit vector in the direction of the satellites velocity. This velocity change vector is then added to the satellites velocity vector, to calculate the new velocity vector.

3. Calculate new elements.

   The orbital elements are calculated with the new velocity vector and the satellite's position vector.

4. Complete the orbit.

   The program will complete the orbit to the new perigee point using the satellite's position, new velocity and new elements. There are a number of problems that arise if the satellite is just stepped around to the perigee point. For example, with velocity changes in the orbital plane the apogee and perigee directions can physically swap. This is a problem when plotting with the perifocal coordinate system because the $X_\omega$ axis points toward perigee. To avoid problems like this the arrays used in plotting the orbit must be cleared and the satellite's current position

and velocity be treated as initial conditions. However; to compare the old and new orbits there is a desire to retain as much of the previous orbit as possible. The velocity changes where divided into the following four cases to handle these problems:

a. Change velocity in the orbital plane at the perigee point with the new velocity greater than the circular velocity. The perigee point will remain the same so the satellite is stepped around using the unperturbed subroutines.

b. Change velocity in the orbital plane at the perigee point with the new velocity less than or equal to the circular velocity. The perigee and apogee directions will switch so the plotting arrays are first cleared and stored with the current location data. Because the satellite is now at the apogee point the satellite is stepped around to the perigee point storing the second half of the orbit. The entire next orbit is calculated and stored to get a complete orbit.

c. Change velocity in the orbital plane at the apogee point with the new velocity less than the circular velocity. The perigee and apogee directions will remain the same, so the satellite is stepped around to the perigee point completing the orbit.

d. This last case catches all the following velocity changes; velocity change in the orbital plane at the apogee point with the new velocity greater than or equal to the circular velocity, velocity changes at any other true anomaly in the orbital plane, and any velocity change out of the orbital plane. The plotting arrays are cleared and stored with the current location data. No matter where in the orbit the satellite is, the satellite is first stepped around to the perigee point, and to ensure a complete orbit is plotted the entire next orbit is also calculated and stored.

## VI. GRAPHICAL PLOTS

The program provides two types of graphical displays of the orbit, a display in the perifocal coordinate system and a display of the satellite's ground track. Each display type is useful in observing different aspects of the orbit. The perifocal display will allow the user to see how certain orbital parameters change with different initial positions and velocities, and also how the parameters change with velocity changes at varying positions in the orbit. The ground track will enable the user to gain an appreciation for the physical location of the satellite above the earth, and see how the orbital parameter affects the path of the satellite. The ground track will also display the precession of a sequence of orbits. Both displays plot the position steps to give the user an understanding of how the satellite speeds up at perigee and slows down around apogee.

The DISSPLA package on the mainframe computer was used to enable the plotting of the orbits. The versatility of plotting subroutines of DISSPLA makes the actual programming of the orbit a simple matter of initializing DISSPLA for the type of monitor being used, setting up the plotting area, initializing the axis and axis scale, and then plotting the desired curve from points contained in arrays. This is a simplified explanation of DISSPLA, but for further details on DISSPLA programming refer to the DISSPLA user's manual [Ref. 5]. DISSPLA also supplies subroutines to draw a variety of projections of the world and fill the projections with coast lines, latitude lines and longitude lines. There are a couple of DISSPLA requirements that did require special handling in the program. The requirement that the data be supplied in arrays forced the program to load arrays with the required position and parameters and to keep a counter for the number in the arrays. The array format requires the size of the array be specified in the beginning of the program. The array size needs to be large enough to hold a number of orbits, but not so large as to waist storage space. The program will continue to add orbital data to the arrays until the user chooses to delete the previous orbits. If a new initial position and velocity is entered or if the arrays will overflow with the next orbit the arrays will automatically delete all previous orbits. DISSPLA also requires that all data be in single precision format. The program calculates all orbits in double precision in order to limit the effect of round-off error, but by using the single precision data for plotting will not affect the accuracy of the plot in any way.

The subroutines used to display the orbits will be covered in the following three sections:

## A. PERIFOCAL PLOT

The plotting of the orbit in the perifocal coordinate system is the easier of the two types of plots. Since the perifocal coordinate system has the orbital plane as the fundamental plane, the only requirements to describe the orbit in the perifocal coordinate system are arrays with the true anomaly and the radius to the satellite. To give the user a sense of the size of the plot, the axis length varies with the eccentricity and semi-major axis length. Also a plot of the earth is plotted to the same scale, with the pole or center of the plot on the origin of the axis. The latitude of the earth at the center of the plot will vary with the inclination of the orbit. This plot will allow the user to see a relative view of the satellite's coverage in the minus 'Z' axis direction of the perifocal coordinate system.

## B. GROUND TRACK

The ground track plot is a very complex subroutine compared with the perifocal plot. Because the ground track is not a continuous curve a procedure to handle the satellite ending at one end of the plot and wrapping around to the other end was developed. The wrap around problem is avoided in most orbits by plotting the orbit in segments with the following two rules. Each segment begins at the beginning of a new plot or at the edge of the plot area, and ending when the satellite would wrap around to the other side of the plot. At the beginning of a segment if the position of the satellite is within five degrees of the edge of the plot, that position and any other positions within that five degree boundary will not be plotted. The segment will end when the satellite is within ten degrees of the edge of the plot. The above restrictions imposed on the segments of the plot will not substantially affect the interpretation or usefulness of the plot. The ground track is plotted on top of a cylindrical equidistant projection of the world, with the world coast lines and a longitude-latitude grid for reference.

## C. DATA

Information concerning the orbit is displayed on the lower half of the plot. The information is designed to supply the user with enough of the basic orbital elements and other parameters affecting the orbit to be able to evaluate what basic type of orbit the satellite is in, and the effects of velocity changes and perturbing forces have on the orbit. The following data are plotted: inclination(i), semi-major axis (a), eccentricity (e), period

(per), apogee and perigee velocity and radius, average time rate-of-change of orbital elements, and the average magnitude of perturbing forces per unit mass.

## VII. CONCLUSIONS AND RECOMMENDATIONS

The program supplies the student with an interactive tool to study the orbital motion of satellites around the earth. The student can investigate a variety of orbits by varying the orbital parameters, command velocity changes, and observe the effects of perturbing forces.

The student is provided with two options for entering the initial position and velocity of the satellite. The program could be expanded to provide the student with the additional options of entering either orbital parameters or a ground observation data and have the program calculate the initial position and velocity from this data. Also the student is limited to orbits with eccentricities less than one (elliptic orbits). The program could be also be expanded to include more eccentric orbit for Lunar, interplanetary, and missile trajectories. The perturbing orbit is calculated for orbits around the earth with relatively small perturbing forces in relation to the earths gravitational force. This fact will cause the program to produce false results if the student tries to calculate lunar trajectories. Special routines would have to be employed when the perturbing force (the moons gravitational attraction) is comparable to the earths gravitational attraction. This will not become a factor for studying current satellite orbits out to the geosynchronous radius of 42241.1km.

The velocity change subroutines move the satellite to a location close to the desired location before a velocity change is imposed. By reducing the step size in the velocity change subroutine, this error could be reduced. Precise orbital transfer maneuvers can be modeled by reducing this error caused by the positioning of the satellite prior to changing the velocity. The program will currently provide the student with useful plots for gaining experience with various transfer orbits by varying the magnitude and location of the velocity changes.

The output of the calculations of the orbit are arrays loaded with the satellite's position and select orbital parameters. The DISSPLA subroutines that plot the points are not unique. The program would become portable to personal computers with these graphics subroutines written in FORTRAN and included in the program.

A final recommendation is that the display of the ground track could be modified to show ground coverage, number of satellites in a constellation, and other elements necessary for planning a real-world artificial satellite application.

# APPENDIX A. ORBIT PROGRAM

```
      PROGRAM ORBIT
*        THIS PROGRAM IS AN INTERACTIVE TIME STEP SIMULATION OF
*     SATELLITES AROUND THE EARTH.  PERTURBED AND UNPERTURBED ORBITS
*     ARE CALCULATED AND PLOTTED.  VELOCITY CHANGES ARE ALSO PERMITTED
*     AT SPECIFIED TRUE ANOMALIES.

*     A LIST OF VARIABLES USED BY THE MAIN PROGRAM FOLLOWS.
*     A     = SEMI-MAJOR AXIS
*     AL    = ARGUMENT OF LONGITUDE
*     AP    = ARGUMENT OF PERIGEE
*     CHTA  = VELOCITY CHANGE LOCATION TRUE ANOMALY
*     DT    = TIME STEP
*     E     = ECCENTRICITY
*     EA    = ECCENTRIC ANOMALY
*     EI    = I VECTOR OF ECCENTRICITY
*     EJ    = J VECTOR OF ECCENTRICITY
*     EK    = K VECTOR OF ECCENTRICITY
*     FR    = R VECTOR OF TOTAL FORCE
*     FS    = S VECTOR OF TOTAL FORCE
*     FW    = W VECTOR OF TOTAL FORCE
*     H     = ANGULAR MOMENTUM
*     HI    = I VECTOR OF ANGULAR MOMENTUM
*     HJ    = J VECTOR OF ANGULAR MOMENTUM
*     HK    = K VECTOR OF ANGULAR MOMENTUM
*     I     = INCLINATION
*     IOPT1= PERTURBED OR UNPERTURBED OPTION
*     IOPT2= OPTIONS: PLOT NEXT ORBIT, CHANGE INITIAL VALUES,
*     CHANGE VELOCITY, PLOT ANOTHER VIEW OF ORBIT, QUIT
*     LAN   = LONGITUDE OF ASCENDING NODE
*     LP    = LONGITUDE OF PERIGEE
*     MA    = MEAN ANOMALY
*     MM    = MEAN MOTION
*     MU    = GRAVITATIONAL PARAMETER
*     N     = ASCENDING NODE
*     NI    = I VECTOR OF ASCENDING NODE
*     NJ    = J VECTOR OF ASCENDING NODE
*     NK    = K VECTOR OF ASCENDING NODE
*     NUM   = STEP COUNTER
*     P     = SEMI-LATUS RECTUM
*     PER   = PERIOD OF ORBIT
*     PI    = PI
*     RA    = RADIUS OF APOGEE
*     RE    = RADIUS OF EARTH
*     R     = ORBITAL RADIUS
*     RI    = I VECTOR OF ORBITAL RADIUS
*     RJ    = J VECTOR OF ORBITAL RADIUS
*     RK    = K VECTOR OF ORBITAL RADIUS
*     T     = TIME COUNTER IN ORBIT
*     TA    = TRUE ANOMALY
*     TDA   = TOTAL CHANGE IN SEMI-MAJOR AXIS
*     TDAP  = TOTAL CHANGE IN ARGUMENT OF PERIGEE
```

```
*     TDE   = TOTAL CHANGE IN ECCENTRICITY
*     TDH   = TOTAL CHANGE IN ANGULAR MOMENTUM
*     TDI   = TOTAL CHANGE IN INCLINATION
*     TDMA  = TOTAL CHANGE IN MEAN ANOMALY
*     TDMM  = TOTAL CHANGE IN MEAN MOTION
*     TDLAN= TOTAL CHANGE IN LONGITUDE OF ASCENDING NODE
*     TF    = TIME OF FLIGHT
*     TFDRA= TOTAL FORCE OF DRAG
*     TFEA = TOTAL FORCE OF EARTH'S OBLATENESS
*     TFMO = TOTAL FORCE FROM MOON
*     TFSU = TOTAL FORCE FROM SUN
*     TL    = TRUE Longitude AT EPOCH
*     TT    = TRUE TIME SINCE SATELLITE HAS BEEN IN ORBIT
*     V     = SATELLITE VELOCITY
*     VI    = I VECTOR OF SATELLITE VELOCITY
*     VJ    = J VECTOR OF SATELLITE VELOCITY
*     VK    = K VECTOR OF SATELLITE VELOCITY

*     A LIST OF THE ARRAYS USED FOLLOWS:

*     AINRAY  = INCLINATION
*     APRAY   = ARGUMENT OF PERIGEE
*     RARAY   = RADIUS
*     RIRAY   = I VECTOR OF RADIUS
*     RJRAY   = J VECTOR OF RADIUS
*     RKRAY   = K VECTOR OF RADIUS
*     TARAY   = TRUE ANOMALY
*     TIMRAY  = TIME

*     A LIST OF SUBROUTINES CALLED BY THE MAIN PROGRAM WILL FOLLOW:

*     CALCEL  = CALCULATES THE ORBITAL ELEMENTS
*     CHGVEL  = ALLOW THE USER TO CHANGE THE VELOCITY OF THE SATELLITE
*     INPUTS  = PROMPTS USER FOR INITIAL POSITION AND VELOCITY
*     INTSUM  = INITIALIZES THE SUMS IN THE ARRAYS
*     NEWELT  = CALCULATE NEW ORBITAL ELEMENTS FROM TIME STEP
*     NEWPOS  = CALCULATE NEW POSITION VECTOR
*     NEWVEL  = CALCULATE NEW VELOCITY VECTOR
*     OPTION  = GIVE THE USER THE OPTIONS Permitted IN THE PROGRAM
*     PLOTS   = PLOTS THE ORBITS
*     PRETUR  = CALCULATES THE PERTURBED ORBIT
*     STORE   = STORE THE POSITION DATA IN ARRAYS
*     UNPRET  = CALCULATE THE UNPERTURBED ORBIT

*     BEGIN MAIN PROGRAM

      DOUBLE PRECISION  PI,MU,RI,RJ,RK,R,VI,VJ,VK,V,HI,HJ,HK,H,
     +   NI,NJ,NK,N,P,EI,EJ,EK,E,A,I,LAN,AP,TA,AL,LP,TL,PER,EA,
     +   MM,MA,T,DT,TF, FR,FS,FW,TT,CHTA,RA,VA,TEMPTA,RE

      DIMENSION TARAY(500),RARAY(500),RIRAY(500),RJRAY(500),RKRAY(500),
     +   AINRAY(500),APRAY(500),TIMRAY(500)

      CHARACTER*1,LOOP,YORN,ORLOOP

      PI = 3.141592653589794
```

```
      MU = 3.986012D+05
      RE = 6.378145D+03

*     USER INTRO TO PROGRAM
      CALL INTRO

*     ENTERED MAIN PROGRAM LOOP
      LOOP = 'Y'
 10   IF (LOOP .EQ. 'Y') THEN

*        INITIALIZE STEP COUNTER AND TRUE TIME
 20      NUM = 1
         TT = 0.0

*        PROMPT USER FOR INITIAL POSITION AND VELOCITY
         CALL INPUTS(RI,RJ,RK,R,VI,VJ,VK,V,MU,LOOP,PI)

*        EXIT PROGRAM
         IF (LOOP .EQ. 'N') THEN
              GOTO 10
         ENDIF

*        CALCULATE AND STORE ORBITAL ELEMENTS
         CALL CALCEL(RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,A,I,LAN,
     +               LP,TA,PER,EA,MA,AP,AL,TF,P,PI,MU,MM,N,H,HI,HJ)
         CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,RARAY,TARAY,
     +              NUM,I,AP,AINRAY,APRAY,TT,TIMRAY)

*        PRINT DATE FOR USER TO REVIEW
         PRINT*,'VI =', VI,' KM/S'
         PRINT*,'VJ =', VJ,' KM/S'
         PRINT*,'VK =', VK,' KM/S'
         PRINT*,' V =', V,' KM/S'
         PRINT*,'RI =', RI,' KM'
         PRINT*,'RJ =', RJ,' KM'
         PRINT*,'RK =', RK,' KM'
         PRINT*,' R =', R,' KM'
         PRINT*,'ECCENTRICITY =',E
         DEGI = SNGL((180.0/PI)*I)
         PRINT*,'INCLINATION =',DEGI,' DEGREES'
         PERHRS = SNGL(PER/3600.0)
         PRINT*,'PERIOD =',PERHRS,' HOURS'
         PRINT*,'ARE THESE VALUES CORRECT?  ENTER "Y" OR "N" :'
         READ*,YORN
         CALL EXCMS('CLRSCRN')
         IF (.NOT. YORN .EQ. 'Y') THEN
              GOTO 20
         ENDIF

*        CALCULATE TIME STEP AND SET TIMER TO ONE TIME STEP
         DT = PER/50
         T = DT

*        STEP SATELLITE TO PERIGEE POINT AND RECORD
 50      IF ((TA.GT.0.063).AND.(TA.LT.6.21)) THEN
              TT = TT + DT
```

```
            CALL NEWELT(MM,MA,E,EA,TA,TF,DT,PI,PER)
            CALL NPOS(RI,RJ,RK,R,LAN,AP,I, TA,A,E)
            CALL NVEL(E,P,TA,LAN,AP,I,VI,VJ,VK,V,MU)
            NUM = NUM + 1
            CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,RARAY,TARAY,
     +           NUM,I,AP,AINRAY,APRAY,TT,TIMRAY)
            T = T + DT
            GOTO 50
         ENDIF

*        CALCULATE ELEMENTS FROM PERIGEE POINT
         CALL CALCEL(RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,A,I,LAN,
     +              LP,TA,PER,EA,MA,AP,AL,TF,P,PI,MU,MM,N,H,HI,HJ)

         DT = PER/50
         T = DT

*        STORE FIRST Unperturbed ORBIT
         CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +              MU,PI,H,A,E,N,TA,P,MM,MA,EA,TF,T,NUM,RIRAY,RJRAY,
     +              RKRAY,RARAY,TARAY,AINRAY,APRAY,TIMRAY,TT)

*        INITIALIZE SUMS FOR FORCE AND ORBITAL ELEMENT CHANGES TO ZERO
         CALL INTSUM(TFEA,TFSU,TFMO,TFDRA,TDI,TDA,TDE,TDMM,TDMA,TDLAN,
     +              TDH,TDAP)

*        PLOT FIRST UNPERTURBED ORBIT
 70      CALL PLOTS(RIRAY,RJRAY,RKRAY,RARAY,TARAY,NUM,PI,I,LP,A,E,TF,
     +             AINRAY,APRAY,TIMRAY,TFEA,TFSU,TFMO,TFDRA,PER,TDI,TDA,
     +             TDE,TDMM,TDMA,TDLAN,TDH,TDAP,MM,MA,LAN,H,AP,R,V)

*        BEGIN NEW ORBIT OPTIONS
*        IOPT1 = 1. Unperturbed ORBIT
*              = 2. Perturbed ORBIT
*              = 3. QUIT
*        IOPT2 = 1. PLOT NEXT ORBIT
*              = 2. CHANGE INITIAL VALUES
*              = 3. CHANGE VELOCITY AT A SPECIFIC TRUE Anomaly
*              = 4. PLOT ANOTHER VIEW OF SAME ORBIT
*
*        ALSO ASKED IF WANT TO CLEAR ALL PREVIOUS ORBITS

*        CALCULATE ELEMENTS AT PERIGEE
 80      CALL CALCEL(RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,A,I,LAN,
     +              LP,TA,PER,EA,MA,AP,AL,TF,P,PI,MU,MM,N,H,HI,HJ)

*        CHECK FOR POSSIBLE ARRAY OVERFLOW
         IF (NUM .GT. 425) THEN
            PRINT*,'ARRAYS ARE FULL'
            PRINT*,'PREVIOUS ORBITS WILL BE ERASED!'
            NUM = 1
            CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,RARAY,TARAY,
     +                NUM,I,AP,AINRAY,APRAY,TT,TIMRAY)
         ENDIF

*        PROMPT USER FOR DESIRED OPTION
```

```
      CALL OPTION(IOPT1,IOPT2,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +            TARAY,AINRAY,APRAY,TIMRAY)

*     Initialize SUMS FOR FORCE AND ORBITAL ELEMENT CHANGES TO ZERO
      CALL INTSUM(TFEA,TFSU,TFMO,TFDRA,TDI,TDA,TDE,TDMM,TDMA,TDLAN,
     +            TDH,TDAP)

*     SET TIME COUNTER TO ONE TIME STEP
      T = DT

*     OPTION: PLOT THE NEXT ORBIT
      IF (IOPT2 .EQ. 1) THEN

*        CALCULATE AND PLOT UNPERTURBED ORBIT
         IF(IOPT1 .EQ. 1) THEN
            CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,
     +           RK,R,VI,VJ,VK,V,MU,PI,H,A,
     +           E,N,TA,P,MM,MA,EA,TF,T,NUM,RIRAY,RJRAY,RKRAY,
     +           RARAY,TARAY,AINRAY,APRAY,TIMRAY,TT)
            CALL PLOTS(RIRAY,RJRAY,RKRAY,RARAY,TARAY,NUM,
     +           PI,I,LP,A,E,TF,AINRAY,APRAY,TIMRAY,
     +           TFEA,TFSU,TFMO,TFDRA,PER,
     +           TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP,
     +           MM,MA,LAN,H,AP,R,V)

*        CALCULATE AND PLOT PERTURBED ORBIT
         ELSEIF(IOPT1 .EQ. 2) THEN
            CALL PRETUR(DT,PER,AL,LAN,AP,I,
     +             RI,RJ,RK,R,VI,VJ,VK,V,FR,FS,FW,
     +             MU,PI,H,A,E,N,TA,P,MM,MA,EA,TF,T,NUM,
     +             RIRAY,RJRAY,RKRAY,RARAY,TARAY,AINRAY,APRAY,
     +             TIMRAY,TT,TFEA,TFSU,TFMO,TFDRA,
     +             TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP)
            CALL PLOTS(RIRAY,RJRAY,RKRAY,RARAY,TARAY,NUM,
     +           PI,I,LP,A,E,TF,AINRAY,APRAY,TIMRAY,
     +           TFEA,TFSU,TFMO,TFDRA,PER,
     +           TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP,
     +           MM,MA,LAN,H,AP,R,V)
         ENDIF

*     GOTO THE BEGINNING OF THE PROGRAM TO CHANGE THE INITIAL VALUES
      ELSEIF (IOPT2 .EQ. 2) THEN
         GOTO 20

*     CHANGE VELOCITY AT A SPECIFIC TRUE ANOMALY AND
*     PLOT THE NEW ORBIT
      ELSEIF (IOPT2 .EQ. 3) THEN
         CALL CHGVEL(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,
     +                 VI,VJ,VK,V,MU,PI,
     +                 H,A,E,N,TA,P,MM,MA,EA,TF,T,NUM,RIRAY,
     +                 RJRAY,RKRAY,RARAY,TARAY,AINRAY,APRAY,
     +                 TIMRAY,TT,EI,EJ,EK,LP,HI,HJ,IOPT1,
     +                 TFEA,TFSU,TFMO,TFDRA,TDI,TDA,TDE,TDMM,
     +                 TDMA,TDLAN,TDH,TDAP)
         CALL PLOTS(RIRAY,RJRAY,RKRAY,RARAY,TARAY,NUM,
     +           PI,I,LP,A,E,TF,AINRAY,APRAY,TIMRAY,
```

```
     +                 TFEA,TFSU,TFMO,TFDRA,PER,
     +                 TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP,
     +                 MM,MA,LAN,H,AP,R,V)

*        PLOT ANOTHER VIEW OF THE SAME ORBIT
         ELSEIF (IOPT2 .EQ. 4) THEN
              CALL PLOTS(RIRAY,RJRAY,RKRAY,RARAY,TARAY,NUM,
     +                 PI,I,LP,A,E,TF,AINRAY,APRAY,TIMRAY,
     +                 TFEA,TFSU,TFMO,TFDRA,PER,
     +                 TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP,
     +                 MM,MA,LAN,H,AP)

*        STOP THE PROGRAM
         ELSEIF (IOPT2 .EQ. 5) THEN
              GOTO 90
         ELSE
              PRINT*,'INVALID ENTRY!'
              GOTO 80
         ENDIF

*        CHECK IF SATELLITE Impacted THE EARTH AND GO TO THE BEGINNING
         IF (R .LE. 6450.0) THEN
              PRINT*,'SATELLITE WILL IMPACT THE EARTH!!!'
              PRINT*,'PROGRAM WILL RESET TO THE BEGINNING'
              GOTO 20
         ENDIF

*        GOTO THE TOP OF THE OPTION LOOP
         GOTO 80

*        GIVE THE USER A CHANCE TO RECOVER THE PROGRAM
 90      PRINT*,'THIS IS YOUR LAST CHANCE!'
         PRINT*,'DO YOU WANT TO CONTINUE?'
         PRINT*,'AND GOTO THE Beginning OF THE PROGRAM?'
         PRINT*,'ENTER "Y" OR "N" :'
         READ*,LOOP
         PRINT*,LOOP
         GOTO 10
      ENDIF

*     DISSPLA SUBROUTINE TO TELL GRAPHICS TERMINAL PLOTTING
*     SESSION IS DONE
      CALL DONEPL
      STOP
      END

*****************************************

      SUBROUTINE INTRO
*     THIS SUBROUTINE WILL GIVE THE USER A Brief INTRODUCTION OF THE
*     USES OF THE PROGRAM

      PRINT*,'THIS PROGRAM IS A GRAPHICS DISPLAY OF Satellite ORBITS.'
      PRINT*,'YOU WILL BE ASKED TO INPUT THE INITIAL VELOCITY AND'
      PRINT*,'POSITION VECTORS OF THE Satellite.  THE PROGRAM WILL  '
      PRINT*,'THEN CALCULATE THE ORBITAL PARAMETERS AND THE '
```

```
      PRINT*,'Unperturbed ORBIT.  THE USER WILL THEN HAVE THE'
      PRINT*,'CHOICE OF DISPLAYS:'
      PRINT*,'     -PERIFICAL (SHOWS RELATIVE SIZE OF ORBIT)'
      PRINT*,'     -Equatorial (SHOWS ORBIT INCLINED, USER INPUT'
      PRINT*,'                 LONGITUDE TO VIEW AT)'
      PRINT*,'     -GROUND TRACK'
      PRINT*,' '
      PRINT*,'THE USER IS THEN ASKED TO CHOOSE ONE OF THE FOLLOWING:'
      PRINT*,'     -Unperturbed ORBITS'
      PRINT*,'     -Perturbed ORBITS'
      PRINT*,'     -VELOCITY CHANGES'
      PRINT*,'THE USER''S CHOICE WILL BE USED IN DEVELOPING THE'
      PRINT*,'GRAPHICAL OUTPUT.'
      PRINT*,' '
      PRINT*,'THE USER IS THEN GIVEN THE FOLLOWING CHOICES:'
      PRINT*,'      -CLEAR ALL THE PREVIOUS ORBITS'
      PRINT*,'      -CHANGE THE INITIAL PARAMETERS'
      PRINT*,'      -CHANGE VELOCITY AT A SPECIFIC TRUE Anomaly'
      PRINT*,'      -PLOT ANOTHER VIEW OF THE SAME ORBIT'
      RETURN
      END

****************************************************************

      SUBROUTINE OPTION(IOPT1,IOPT2,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +                  TARAY,AINRAY,APRAY,TIMRAY)
*     THIS SUBROUTINE GIVES THE USER A CHOICE OF OPERATIONS THAT CAN BE
*       PERFORMED ON THE PROGRAM AND RETURNS THE USERS CHOICE WITH
*       VARIABLES IOPT1 AND IOPT2

      DIMENSION RIRAY(500),RJRAY(500),RKRAY(500),RARAY(500),TARAY(500),
     +   AINRAY(500),APRAY(500),TIMRAY(500)
      CHARACTER*1,YORN
      IOPT1 = 0

*     PROMPT USER FOR OPTION
  103 PRINT*,'WHICH OF THE FOLLOWING OPTIONS WOULD YOU LIKE:'
      PRINT*,'   1.-CALCULATE THE NEXT ORBIT USING THE SAME'
      PRINT*,'      PARAMETERS'
      PRINT*,'   2.-CHANGE THE INITIAL PARAMETERS OF THE ORBIT'
      PRINT*,'   3.-CHANGE THE VELOCITY AT A POINT IN THE ORBIT'
      PRINT*,'      (THE UNPERTURBED ORBIT WILL BE USED)'
      PRINT*,'   4.-PLOT ANOTHER VIEW OF THE ORBIT(S)'
      PRINT*,'   5.-QUIT'
      PRINT*,'ENTER 1, 2, 3, 4, OR 5:'
      READ*,IOPT2
      PRINT*,IOPT2
      CALL EXCMS('CLRSCRN')
      IF ( IOPT2 .GT. 5) THEN
         GOTO 103
      ENDIF

*     Prompt USER FOR TYPE OF ORBIT DESIRED
  105 IF (IOPT2 .EQ. 1) THEN
         PRINT*,'WHICH TYPE OF ORBIT WOULD YOU LIKE TO SEE,'
         PRINT*,'    1.-Unperturbed ORBITS'
```

```
          PRINT*,'    2.-Perturbed ORBITS'
          PRINT*,' ENTER 1 OR 2:'
          READ*,IOPT1
          PRINT*,IOPT1
          CALL EXCMS('CLRSCRN')
          IF ((IOPT1 .NE. 1) .AND. (IOPT1 .NE. 2)) THEN
               PRINT*,'INVALID ENTRY!'
               GOTO 105
          ENDIF
      ENDIF

*     PROMPT USER TO CLEAR PREVIOUS ORBITS
 107  IF ((IOPT2 .EQ. 1) .OR. (IOPT2 .EQ. 3)) THEN
          PRINT*,'DO YOU WANT TO CLEAR THE PREVIOUS ORBITS?'
          PRINT*,'ENTER "Y" OR "N" :'
          READ*,YORN
          PRINT*,YORN
          CALL EXCMS('Clrscrn')
          IF (YORN .EQ. 'Y') THEN
               RIRAY(1) = RIRAY(NUM)
               RJRAY(1) = RJRAY(NUM)
               RKRAY(1) = RKRAY(NUM)
               RARAY(1) = RARAY(NUM)
               TARAY(1) = TARAY(NUM)
               AINRAY(1) = AINRAY(NUM)
               APRAY(1) = APRAY(NUM)
               TIMRAY(1) = TIMRAY(NUM)
               NUM = 1
          ELSEIF (YORN .NE. 'N') THEN
               PRINT*,'INVALID ENTRY!!'
               PRINT*,'ALL INPUTS MUST BE CAPITOL LETTERS'
               GOTO 107
          ENDIF
      ENDIF

* CHECK FOR INVALID OPTION
      IF ((IOPT2 .NE. 1).AND. (IOPT2 .NE. 2).AND. (IOPT2 .NE. 3) .AND.
     +     (IOPT2 .NE. 4).AND. (IOPT2 .NE. 5)) THEN
          PRINT*,'INVALID ENTRY!'
          GOTO 103
      ENDIF
      RETURN
      END

*****************************************************************
* COORDINATE TRANSFORMATIONS
*****************************************************************

      SUBROUTINE PQWIJK(LAN,AP,INC,P,Q,W,I,J,K)
*     THIS SUBROUTINE TRANSFORMS PQW COORDINATES TO IJK COORDINATES

      DOUBLE PRECISION  INC,P,Q,W,I,J,K,R11,R12,R13,R21,R22,R23,
     +    R31,R32,R33,LAN,AP
      R11 = DCOS(LAN)*DCOS(AP) - DSIN(LAN)*DSIN(AP)*DCOS(INC)
      R12 = -DCOS(LAN)*DSIN(AP) - DSIN(LAN)*DCOS(AP)*DCOS(INC)
      R13 = DSIN(LAN)*DSIN(INC)
```

```
      R21 = DSIN(LAN)*DCOS(AP) + DCOS(LAN)*DSIN(AP)*DCOS(INC)
      R22 = -DSIN(LAN)*DSIN(AP) + DCOS(LAN)*DCOS(AP)*DCOS(INC)
      R23 = -DCOS(LAN)*DSIN(INC)
      R31 = DSIN(AP)*DSIN(INC)
      R32 = DCOS(AP)*DSIN(INC)
      R33 = DCOS(INC)
      I = R11*P + R12*Q + R13*W
      J = R21*P + R22*Q + R23*W
      K = R31*P + R32*Q + R33*W
      RETURN
      END

*********************************************************************

      SUBROUTINE IJKPQW(LAN,AP,INC,I,J,K,P,Q,W)
*     THIS SUBROUTINE TRANSFORMS IJK COORDINATES TO PQW COORDINATES

      DOUBLE PRECISION  INC,I,J,K,P,Q,W,R11,R12,R13,R21,R22,R23,
     +   R31,R32,R33,LAN,AP
      R11 = DCOS(LAN)*DCOS(AP) - DSIN(LAN)*DSIN(AP)*DCOS(INC)
      R21 = -DCOS(LAN)*DSIN(AP) - DSIN(LAN)*DCOS(AP)*DCOS(INC)
      R31 = DSIN(LAN)*DSIN(INC)
      R12 = DSIN(LAN)*DCOS(AP) + DCOS(LAN)*DSIN(AP)*DCOS(INC)
      R22 = -DSIN(LAN)*DSIN(AP) + DCOS(LAN)*DCOS(AP)*DCOS(INC)
      R32 = -DCOS(LAN)*DSIN(INC)
      R13 = DSIN(AP)*DSIN(INC)
      R23 = DCOS(AP)*DSIN(INC)
      R33 = DCOS(INC)
      P = R11*I + R12*J + R13*K
      Q = R21*I + R22*J + R23*K
      W = R31*I + R32*J + R33*K
      RETURN
      END

*********************************************************************

      SUBROUTINE IJKRSW(LAN,AL,INC,I,J,K,R,S,W)
*     THIS SUBROUTINE CHANGES FROM IJK COORDINATES TO RSW COORDINATES

      DOUBLE PRECISION  INC,I,J,K,R,S,W,R11,R12,R13,R21,R22,R23,
     +   R31,R32,R33,LAN,AL
      R11 = DCOS(LAN)*DCOS(AL) - DSIN(LAN)*DCOS(INC)*DSIN(AL)
      R12 = DSIN(LAN)*DCOS(AL) + DSIN(AL)*DCOS(LAN)*DCOS(INC)
      R13 = DSIN(INC)*DSIN(AL)
      R21 = -DCOS(LAN)*DSIN(AL)-DSIN(LAN)*DCOS(INC)*DCOS(AL)
      R22 = -DSIN(LAN)*DSIN(AL) + DCOS(LAN)*DCOS(INC)*DCOS(AL)
      R23 = DSIN(INC)*DCOS(AL)
      R31 = DSIN(LAN)*DSIN(INC)
      R32 = -DCOS(LAN)*DSIN(INC)
      R33 = DCOS(INC)
      R = R11*I + R12*J + R13*K
      S = R21*I + R22*J + R23*K
      W = R31*I + R32*J + R33*K
      RETURN
      END
```

```
*************************************************************************

      SUBROUTINE RSWIJK(LAN,AL,INC,R,S,W,I,J,K)
*     THIS SUBROUTINE CHANGES FROM RSW COORDINATES TO IJK COORDINATES

      DOUBLE PRECISION  INC,R,S,W,I,J,K,R11,R12,R13,R21,R22,R23,
     +   R31,R32,R33,LAN,AL
      R11 = DCOS(LAN)*DCOS(AL) - DSIN(LAN)*DCOS(INC)*DSIN(AL)
      R21 = DSIN(LAN)*DCOS(AL) + DSIN(AL)*DCOS(LAN)*DCOS(INC)
      R31 = DSIN(INC)*DSIN(AL)
      R12 = -DCOS(LAN)*DSIN(AL)-DSIN(LAN)*DCOS(INC)*DCOS(AL)
      R22 = -DSIN(LAN)*DSIN(AL) + DCOS(LAN)*DCOS(INC)*DCOS(AL)
      R32 = DSIN(INC)*DCOS(AL)
      R13 = DSIN(LAN)*DSIN(INC)
      R23 = -DCOS(LAN)*DSIN(INC)
      R33 = DCOS(INC)
      I = R11*R + R12*S + R13*W
      J = R21*R + R22*S + R23*W
      K = R31*R + R32*S + R33*W
      RETURN
      END

*************************************************************************

      SUBROUTINE PQWRSW(TA,P,Q,W,R,S,WN)
*     THIS SUBROUTINE CHANGES FROM PQW COORDINATES TO RSW COORDINATES

      DOUBLE PRECISION  P,Q,W,R,S,WN,R11,R12,R13,R21,R22,R23,
     +   R31,R32,R33,TA
      R11 = DCOS(TA)
      R12 = DSIN(TA)
      R13 = 0.0
      R21 = -DSIN(TA)
      R22 = DCOS(TA)
      R23 = 0.0
      R31 = 0.0
      R32 = 0.0
      R33 = 1.0
      R = R11*P + R12*Q + R13*W
      S = R21*P + R22*Q + R23*W
      WN = R31*P + R32*Q +R33*W
      RETURN
      END

*************************************************************************

      SUBROUTINE RSWPQW(TA,R,S,W,P,Q,WN)
*     THIS SUBROUTINE CHANGES FROM RSW COORDINATES TO PQW COORDINATES

      DOUBLE PRECISION  R,S,W,P,Q,WN,R11,R12,R13,R21,R22,R23,
     +   R31,R32,R33,TA
      R11 = DCOS(TA)
      R21 = DSIN(TA)
      R31 = 0.0
      R12 = -DSIN(TA)
```

```
      R22 = DCOS(TA)
      R32 = 0.0
      R13 = 0.0
      R23 = 0.0
      R33 = 1.0
      P = R11*R + R12*S + R13*W
      Q = R21*R + R22*S + R23*W
      WN = R31*R + R32*S +R33*W
      RETURN
      END

************************************************************
*    STORE ELEMENTS IN ARRAYS
************************************************************

      SUBROUTINE STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,RARAY,TARAY,NUM,
     +   I,AP,AINRAY,APRAY,TT,TIMRAY)
*     THIS SUBROUTINE STORES THE POSITION AND ELEMENTS IN ARRAYS IN
*     SINGLE PRECISION FORM, FOR PLOTTING

      DOUBLE PRECISION RI,RJ,RK,R,TA,I,AP,TT

      DIMENSION RIRAY(500),RJRAY(500),RKRAY(500),RARAY(500),TARAY(500),
     +   AINRAY(500),APRAY(500),TIMRAY(500)

      RIRAY(NUM) = SNGL(RI)
      RJRAY(NUM) = SNGL(RJ)
      RKRAY(NUM) = SNGL(RK)
      RARAY(NUM) = SNGL(R)
      TARAY(NUM) = SNGL(TA)
      AINRAY(NUM) = SNGL(I)
      APRAY(NUM) = SNGL(AP)
      TIMRAY(NUM) = SNGL(TT)
      RETURN
      END

********************************************************************
* INITIAL       POSITION, VELOCITY
********************************************************************

      SUBROUTINE INPUTS(RI,RJ,RK,R,VI,VJ,VK,V,MU,QUIT,PI)
*     THIS SUBROUTINE GIVES THE USER A CHOICE TO EITHER ENTER THE
*     INITIAL POSITION AND VELOCITY VECTOR OR TO LET THE PROGRAM
*     CALCULATE THE INITIAL POSITION AND VELOCITY FROM USER PROMPTED
*     INPUTS

*     SUBROUTINES CALLED FROM THIS SUBROUTINE:
*     INELTS = Prompts USER FOR ORBITAL ELEMENTS
*     IPOS   = PROMPTS USER FOR INITIAL POSITION (IJK)
*     IVEL   = PROMPTS USER FOR INITIAL Velocity (IJK)

      DOUBLE PRECISION RI,RJ,RK,R,VI,VJ,VK,V,MU,PI
      CHARACTER*1,QUIT

*     PROMPT USER FOR METHOD TO ENTER INPUTS
 195  PRINT*,'IN WHICH MANNER WOULD YOU LIKE TO INPUT THE INITIAL'
```

```
      PRINT*,'POSITION AND VELOCITY OF THE SATELLITE?'
      PRINT*,'      1: BY Inputting THE INITIAL POSITION AND VELOCITY'
      PRINT*,'         VECTORS IN THE PERIFOCAL COORDINATE SYSTEM (IJK)'
      PRINT*,'      2: BY LETTING THE SATELLITE BE PLACED ON THE "I"'
      PRINT*,'         AXIS OF THE (IJK) SYSTEM AT A DESIRED RADIUS OF'
      PRINT*,'         PERIGEE(RP) AND INPUTTING EITHER A DESIRED RADIUS'
      PRINT*,'         OF APOGEE(RA), A DESIRED ECCENTRICITY(E), OR THE'
      PRINT*,'         DESIRED VELOCITY AT THAT RADIUS, AND A DESIRED'
      PRINT*,'         INCLINATION(I).'
      PRINT*,'      3: QUIT'
      PRINT*,'ENTER 1, 2 OR 3:'
      READ*,ICHC
      PRINT*,ICHC
      CALL EXCMS('CLRSCRN')

*     USER INPUTS POSITION AND VELOCITY VECTORS
      IF (ICHC .EQ. 1) THEN
         CALL IPOS(RI,RJ,RK,R)
         CALL IVEL(VI,VJ,VK,V,R,MU)

*     USER INPUTS ORBITAL ELEMENTS TO GET POSITION AND VELOCITY
      ELSEIF (ICHC .EQ. 2) THEN
         CALL INELTS(RI,RJ,RK,R,VI,VJ,VK,V,MU,PI)

*     STOP PROGRAM
      ELSEIF (ICHC .EQ. 3) THEN
         QUIT = 'N'
      ELSE
         PRINT*,'INVALID ENTRY!  TRY AGAIN!'
         GOTO 195
      ENDIF
      RETURN
      END

************************************************************************

      SUBROUTINE IPOS(RI,RJ,RK,R)
*     THIS SUBROUTINE ASKS THE USER FOR THE INITIAL POSITION OF THE
*     Satellite IN GEOCENTRIC-EQUATORIAL COORDINATE SYSTEM

      DOUBLE PRECISION RI,RJ,RK,R

      CHARACTER*1, CHOICE
      LOGICAL CORREC
      CORREC  = .FALSE.

*     PROMPT USER FOR VELOCITY VECTOR
 180  IF(.NOT.CORREC) THEN
         CALL EXCMS('CLRSCRN')
         PRINT*,'ENTER RADIUS VECTOR VALUES IN "KM"'
         PRINT*,'RADIUS OF THE EARTH = 6400 KM'
         CORREC  = .TRUE.
         PRINT*,'ENTER RI :'
         READ*,RI
         PRINT*,'RI = ',RI,'KM'
         PRINT*,'ENTER RJ :'
```

```
          READ*,RJ
          PRINT*,'RJ = ',RJ,'KM'
          PRINT*,'ENTER RK :'
          READ*,RK
          PRINT*,'RK = ',RK,'KM'

*         CALCULATE TOTAL R
          R = DSQRT((RI**2) + (RJ**2) + (RK**2))
          PRINT*,'R = ',R,'KM'
          IF (R .LE. 6400.0) THEN
               PRINT*,'RADIUS TO SMALL!!  ENTER NEW VALUES!!'
               GOTO 180
          ENDIF

*         CHECK WITH USER THAT Values ARE CORRECT
          PRINT*,'ARE THESE VALUES CORRECT?'
          PRINT*,'     ENTER "Y" OR "N" :'
          READ*,CHOICE
          CHOICE = 'Y'
          PRINT*,CHOICE
          IF (CHOICE.EQ.'Y') THEN
               CORREC  = .TRUE.
          ENDIF
          GOTO 180
       ENDIF
       RETURN
       END

******************************************************************************

       SUBROUTINE IVEL(VI,VJ,VK,V,R,MU)
*      THIS SUBROUTINE ASKS THE USER FOR THE INITIAL VELOCITY OF THE
*      Satellite

       DOUBLE PRECISION VI,VJ,VK,V,R,VCIR,VMAX,MU

       CHARACTER*1, CHOICE
       LOGICAL CORREC
       CORREC  = .FALSE.

*      CALCULATE ESCAPE VELOCITY AND CIRCULAR VELOCITY AND PROMPT USER
*      FOR VELOCITY VECTOR
 190   IF(.NOT.CORREC) THEN
          CALL EXCMS('CLRSCRN')
          VCIR = DSQRT(MU/R)
          VMAX = DSQRT((2.0*MU)/R)
          PRINT*,'CIRCULAR VELOCITY = ',VCIR,'KM/SEC'
          PRINT*,'MAXIMUM VELOCITY = ',VMAX,'KM/SEC'
          CORREC  = .TRUE.
          PRINT*,'ENTER VELOCITY VECTOR IN (KM/SEC)'

          PRINT*,'ENTER VI :'
          READ*,VI
          PRINT*,'VI = ',VI,'KM/SEC'
          PRINT*,'ENTER VJ :'
          READ*,VJ
```

```
         PRINT*,'VJ = ',VJ,'KM/SEC'
         PRINT*,'ENTER VK :'
         READ*,VK
         PRINT*,'VK = ',VK,'KM/SEC'

*        CALCULATE TOTAL VELOCITY (V)
         V = DSQRT((VI**2) + (VJ**2) + (VK**2))
         PRINT*,'V = ',V,'KM/SEC'

*        CHECK WITH USER THAT VALUES ARE CORRECTS
         PRINT*,'ARE THESE VALUES CORRECT?'
         PRINT*,'     ENTER "Y" OR "N" :'
         READ*,CHOICE
         CHOICE = 'Y'
         PRINT*,CHOICE
         IF (CHOICE.EQ.'Y') THEN
              CORREC  = .TRUE.
         ENDIF
         IF (V .GE. VMAX) THEN
              PRINT*,'VELOCITY IS GREATER THAN THE ESCAPE VELOCITY!!'
              PRINT*,'RE-ENTER VELOCITY!!!'
              CORREC = .FALSE.
         ENDIF
         GOTO 190
      ENDIF
      RETURN
      END

*************************************************************************

      SUBROUTINE INELTS(RI, RJ, RK, R, VI, VJ, VK, V,MU,PI)
*     SATELLITE PLACED ON 'I' AXIS AND USER SUPPLY ORBITAL ELEMENTS TO
*     GET INITIAL POSITION AND VELOCITY

      DOUBLE PRECISION RI,RJ,RK,R,VI,VJ,VK,V,MU,I,ENR,A,E,RP,RA,PI,VMAX
      CHARACTER*1,CHOICE

*     PROMPT USER FOR PERIGEE RADIUS
198   PRINT*,'ENTER RADIUS OF PERIGEE(RP) IN (KM), FOR EXAMPLE:'
      PRINT*,'LOW EARTH ORBIT (LEO), RP = 6600.0 KM'
      PRINT*,'GEOSYNCROUNOUS ORBIT, RP = 42241.1 KM'
      PRINT*,'ENTER RP:'
      PRINT*,'"RP" MUST BE > 6400KM'
      READ*,RP
      PRINT*,RP

*     CHECK FOR VALID RADIUS
      IF (RP .LT. 6400.0) THEN
         PRINT*,'YOUR "RP" IS TO SMALL!!'
         GOTO 198
      ENDIF

*     PROMPT USER FOR TYPE OF INPUT
      PRINT*,'DO YOU WANT TO ENTER THE ECCENTRICITY (E), '
      PRINT*,'RADIUS OF APOGEE (RA), OR VELOCITY (V)?'
      PRINT*,'ENTER "E", "R", OR "V":'
```

```
      READ*,CHOICE
      PRINT*,CHOICE
      CALL EXCMS('CLRSCRN')

*     USER ENTERS Eccentricity AND SEMI-MAJOR AXIS, ENERGY AND VELOCITY
*     IS CALCULATED IN THAT ORDER
      IF (CHOICE .EQ. 'E') THEN
         PRINT*,'ENTER ECCENTRICITY (E):'
         PRINT*,'0.0 <= E < 1.0'
         READ*,E
         PRINT*,E

*        CHECK FOR VALID ECCENTRICITY
         IF ((E .LT. 0.0) .OR. (E .GE. 1.0)) THEN
            PRINT*,'INVALID "E"'
            GOTO 198
         ENDIF
         A = RP/(1-E)
         ENR = -MU/(2.0*A)
         V = DSQRT(2*(ENR+(MU/RP)))

*        USER INPUTS RADIUS OF APOGEE AND ECCENTRICITY IS CALCULATED
*        THEN SEMI-MAJOR AXIS, ENERGY AND THEN VELOCITY.
      ELSEIF (CHOICE .EQ. 'R') THEN
         PRINT*,'ENTER RADIUS OF APOGEE (RA) IN KM:'
         PRINT*,'"RA" MUST BE >="RP", "RP" = ',RP
         READ*,RA
         PRINT*,RA

*        CHECK FOR VALID RADIUS OF APOGEE
         IF (RA .LT. RP) THEN
            PRINT*,'YOUR "RA" IS TO SMALL!!'
            GOTO 198
         ENDIF
         E = (RA-RP)/(RA+RP)
         A = RP/(1-E)
         ENR = -MU/(2.0*A)
         V = DSQRT(2*(ENR+(MU/RP)))

*        USER INPUTS MAGNITUDE OF VELOCITY, PROGRAM PROVIDES CIRCULAR
*        AND ESCAPE VELOCITY FOR COMPARISON AND TO CHECK FOR VALID
*        INPUTS
      ELSEIF (CHOICE .EQ. 'V') THEN
         PRINT*,'ENTER VELOCITY IN KM/SEC:'
         PRINT*,'THE MINIMUM VELOCITY ALLOWED IS FOR A CIRCULAR ORBIT'
         VCIRC = SQRT(SNGL(MU/RP))
         PRINT*,'ORBIT. V(Circular) = ',VCIRC,' KM/S'
         VMAX = DSQRT(2*(MU/RP))
         PRINT*,'THE MAXIMUM VELOCITY < ',VMAX,' KM/S'
         READ*,V
         PRINT*,V
         IF (V .LT. VCIRC) THEN
            PRINT*,'VELOCITY TO SMALL!'
            GOTO 198
         ENDIF
         IF (V .GE. VMAX) THEN
```

```
            PRINT*,'VELOCITY TO GREAT!!'
            GOTO 198
         ENDIF
      ELSE
         PRINT*,'INVALID ENTRY! TRY AGAIN'
         GOTO 198
      ENDIF

*     INCLINATION NEEDED TO GIVE Velocity A Direction
      PRINT*,'ENTER INCLINATION (I) IN DEGREES:'
      READ*,I
      PRINT*,I
      I = (PI/180.0)*I
      VK = V*DSIN(I)
      VJ = V*DCOS(I)
      VI = 0.0

*     RADIUS VECTOR SET
      RI = RP
      RJ = 0.0
      RK = 0.0
      R = RP
      RETURN
      END

***********************************************************************
* CALCULATE THE ORBITAL ELEMENTS
***********************************************************************

      SUBROUTINE CALCEL(RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,A,I,LAN,
     +                 LP,TA,PER,EA,MA,AP,AL,TF,P,PI,MU,MM,N,H,HI,HJ)
*     THIS SUBROUTINE CALLS THE INDIVIDUAL SUBROUTINES TO CALCULATE THE
*     ORBITAL ELEMENTS

*     THIS SUBROUTINE CALLS THE FOLLOWING SUBROUTINES(RETURNED VALUES)
*     ENERGY = ENERGY PER MASS (ENR)
*     ANGMOM = ANGULAR MOMENTUM (H,HI,HJ,HK)
*     NODE   = NODE VECTOR (N,NI,NJ,NK)
*     LATREC = SEMI-LATUS RECTUS (P)
*     ECC    = ECCENTRICITY (E,EI,EJ,EK)
*     SMAXIS = SEMI-MAJOR AXIS (A)
*     INCL   = INCLINATION (I)
*     ASNODE = LONGITUDE OF ASCENDING NODE (LAN)
*     ARP    = ARGUMENT OF PERIGEE (AP)
*     IJKPQW = 'IJK' SYSTEM TO 'PQW' SYSTEM
*     TANOM  = TRUE ANOMALY (TA)
*     ARLAT  = ARGUMENT OF LATITUDE (AL)
*     LONPER = LONGITUDE OF Perigee (LP)
*     TLON   = TRUE LONGITUDE (TL)
*     PERIOD = PERIOD (PER)
*     ECCAN  = ECCENTRIC ANOMALY (EA)
*     MEANMO = MEAN MOTION (MM)
*     MEANAN = MEAN ANOMALY (MA)
*     TFLGHT = TIME OF FLIGHT (TF)

      DOUBLE PRECISION RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,A,I,LAN,AL,
```

```
     +   LP,TA,PER,EA,MA,AP,TF,HI,HJ,HK,H,NI,NJ,NK,N,P,PI,MU,MM,ENR,
     +   TL,RP,RQ,RW,NP,NQ,NW

      CALL ENERGY(V,R,MU,ENR)
      CALL ANGMOM(RI,RJ,RK,VI,VJ,VK,HI,HJ,HK,H)
      CALL NODE(HI,HJ,NI,NJ,NK,N)
      CALL LATREC(H,P,MU)
      CALL ECC(RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,MU)
      CALL SMAXIS(MU,ENR,A)
      CALL INCL(HK,H,I,PI)

*     SPECIAL CASE IF INCLINATION = 0.0
      IF (I.NE.0.0) THEN
         CALL ASNODE(NI,N,LAN,NJ,PI)
         CALL ARP(NI,NJ,N,EI,EJ,EK,E,AP,PI,NP,NQ,LAN)
      ELSE
         LAN = 0.0
         AP = 0.0
      ENDIF

*     COORDINATE TRANSFORMATION OF 'R' AND 'V' VECTORS
      CALL IJKPQW(LAN,AP,I,RI,RJ,RK,RP,RQ,RW)
      CALL IJKPQW(LAN,AP,I,NI,NJ,NK,NP,NQ,NW)
      CALL TANOM(EI,EJ,EK,E,RI,RJ,RK,RP,RQ,RW,R,VI,VJ,VK,TA,PI)

*     SPECIAL CASE FOR Inclination = 0.0
      IF (I .NE. 0.0) THEN
         CALL ARLAT(NI,NJ,NK,N,RI,RJ,RK,R,AL,PI,TA,AP)
      ELSE
         AL = TA
      ENDIF
      CALL LONPER(LAN,AP,LP)
      CALL TLON(LAN,AP,TA,TL)
      CALL PERIOD(A,PER,PI,MU)
      CALL ECCAN(E,TA,EA,PI)
      CALL MEANMO(A,MM,MU)
      CALL MEANAN(EA,E,MA)
      CALL TFLGHT(MM,MA,TF)
      RETURN
      END

**********************************************************************

      SUBROUTINE ENERGY(V,R,MU,ENR)
*     THIS SUBROUTINE CALCULATES THE ENERGY OF THE ORBIT

      DOUBLE PRECISION V,R,MU,ENR

      ENR = ((V**2)/2) - (MU/R)
      RETURN
      END

**********************************************************************

      SUBROUTINE ANGMOM(RI,RJ,RK,VI,VJ,VK,HI,HJ,HK,H)
*     THIS SUBROUTINE CALCULATES THE ANGULAR MOMENTUM
```

```
      DOUBLE PRECISION RI,RJ,RK,VI,VJ,VK,HI,HJ,HK,H

      HI = (RJ * VK) - (RK * VJ)
      HJ = (RK * VI) - (RI * VK)
      HK = (RI * VJ) - (RJ * VI)
      H = DSQRT((HI**2) + (HJ**2) + (HK**2))
      RETURN
      END

*****************************************************************

      SUBROUTINE NODE(HI,HJ,NI,NJ,NK,N)
*     THIS SUBROUTINE CALCULATES THE NODE VECTOR

      DOUBLE PRECISION HI,HJ,NI,NJ,NK,N

      NI = -HJ
      NJ = HI
      NK = 0.0
      N = DSQRT((NI**2) + (NJ**2))
      RETURN
      END

*****************************************************************

      SUBROUTINE LATREC(H,P,MU)
*     THIS SUBROUTINE CALCULATES THE SEMI-LATUS RECTUM

      DOUBLE PRECISION H,P,MU

      P = (H**2)/MU
      RETURN
      END

*****************************************************************

      SUBROUTINE ECC(RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,MU)
*     THIS SUBROUTINE CALCULATES THE ECCENTRICITY

      DOUBLE PRECISION RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,MU,DOT

*     CALCULATE DOT PRODUCT OF 'R' AND 'V' VECTORS
      DOT = (RI*VI) + (RJ*VJ) + (RK*VK)
      EI = (1.0D+00/MU) * (((V**2) - (MU/R)) * RI - (DOT)*VI)
      EJ = (1.0D+00/MU) * (((V**2) - (MU/R)) * RJ - (DOT)*VJ)
      EK = (1.0D+00/MU) * (((V**2) - (MU/R)) * RK - (DOT)*VK)
      E = DSQRT((EI**2) + (EJ**2) + (EK**2))
      RETURN
      END

*****************************************************************

      SUBROUTINE SMAXIS(MU,ENR,A)
*     THIS SUBROUTINE Calculates THE SEMI-MAJOR AXIS

```

```
      DOUBLE PRECISION MU,ENR,A

      A = -MU/(2*ENR)
      RETURN
      END

******************************************************************

      SUBROUTINE INCL(HK,H,I,PI)
*     THIS SUBROUTINE CALCULATES THE INCLINATION
*     'I' ALWAYS LESS THAN 180 DEGREES

      DOUBLE PRECISION HK,H,I,PI

      I = DACOS(HK/H)
      RETURN
      END

******************************************************************

      SUBROUTINE ASNODE(NI,N,LAN,NJ,PI)
*     THIS SUBROUTINE CALCULATES THE LONGITUDE OF THE ASCENDING NODE
*     IF 'NJ' > 0       THEN 'LAN' < 180 DEGREES

      DOUBLE PRECISION NI,N,LAN,NJ,PI

      LAN = DATAN2(NJ,NI)
      IF (LAN .LT. 0.0) THEN
         LAN = (2*PI) + LAN
      ENDIF
      RETURN
      END

******************************************************************

      SUBROUTINE ARP(NI,NJ,N,EI,EJ,EK,E,AP,PI,NP,NQ,LAN)
*     THIS SUBROUTINE CALCULATES THE ARGUMENT OF Perigee
*     IF 'EK' GREATER THAN 0 THEN 'AP' < 180
*     VARIABLE TEMP USED AS A Temporary VALUE FOR ARCTAN

      DOUBLE PRECISION NI,NJ,N,EI,EJ,EK,E,AP,PI,NQ,NP,TEMP,LAN

      IF ((EI .EQ. 0.0) .AND. (EJ .EQ. 0.0)) THEN
         AP = 0.0
      ELSE
         TEMP = DATAN2(EJ,EI)
         IF (TEMP .GT. LAN) THEN
            AP = TEMP - LAN
         ELSE
            AP = (2*PI) - (LAN - TEMP)
          ENDIF
         IF ( AP .LT. 0.0) THEN
            AP = (2*PI) + AP
         ENDIF
         IF (AP .GT. (2*PI)) THEN
            AP = AP - (2*PI)
```

```
         ENDIF
      ENDIF
      RETURN
      END

************************************************************************

      SUBROUTINE TANOM(EI,EJ,EK,E,RI,RJ,RK,RP,RQ,RW,R,VI,VJ,VK,
     +                 TA,PI)
*     THIS SUBROUTINE CALCULATES THE TRUE Anomaly
*     IF (R DOT V) > 0 THEN TA < 180 DEGREES

      DOUBLE PRECISION  DOT,EI,EJ,EK,E,RI,RJ,RK,R,VI,VJ,VK,TA,PI,
     +   RP,RQ,RW

      TA = DATAN2(RQ,RP)
      IF (TA .LT. 0.0 ) THEN
         TA = (2 * PI) + TA
      ENDIF
      RETURN
      END

************************************************************************

      SUBROUTINE ARLAT(NI,NJ,NK,N,RI,RJ,RK,R,AL,PI,TA,AP)
*     THIS SUBROUTINE CALCULATES THE ARGUMENT OF LATITUDE
*     IF (RK > 0) THEN AL < 180 DEGREES

      DOUBLE PRECISION NI,NJ,NK,N,RI,RJ,RK,R,AL,PI,TA,AP

      AL = TA + AP
      RETURN
      END

************************************************************************

      SUBROUTINE LONPER(LAN,AP,LP)
*     THIS SUBROUTINE CALCULATES THE LONGITUDE OF PERIGEE

      DOUBLE PRECISION LAN,AP,LP

      LP = LAN + AP
      RETURN
      END

************************************************************************

      SUBROUTINE TLON(LAN,AP,TA,TL)
*     THIS SUBROUTINE CALCULATES THE TRUE LONGITUDE AT EPOCH

      DOUBLE PRECISION LAN,AP,TA,TL

      TL = AP + LAN + TA
      RETURN
      END
```

```
************************************************************************

      SUBROUTINE PERIOD(A,PER,PI,MU)
*     THIS SUBROUTINE CALCULATES THE PERIOD

      DOUBLE PRECISION A,PER,PI,MU

      PER = 2.0D+00*(PI)*DSQRT((A**3)/MU)
      RETURN
      END

************************************************************************

      SUBROUTINE ECCAN(E,TA,EA,PI)
*     THIS SUBROUTINE CALCULATES THE ECCENTRIC Anomaly

      DOUBLE PRECISION E,TA,EA,PI

      EA = DACOS((E + DCOS(TA))/(1.0D+00 + E*DCOS(TA)))
      IF (TA .GT. PI) THEN
         EA = (2*PI) - EA
      ENDIF
      RETURN
      END

************************************************************************

      SUBROUTINE MEANMO(A,MM,MU)
*     THIS SUBROUTINE CALCULATES THE MEAN MOTION

      DOUBLE PRECISION A,MM,MU

      MM = DSQRT(MU/(A**3))
      RETURN
      END

************************************************************************

      SUBROUTINE MEANAN(EA,E,MA)
*     THIS SUBROUTINE CALCULATES THE MEAN Anomaly

      DOUBLE PRECISION EA,E,MA

      MA = EA - E*DSIN(EA)
      RETURN
      END

************************************************************************

      SUBROUTINE TFLGHT(MM,MA,TF)
*     THIS SUBROUTINE CALCULATES THE TIME OF FLIGHT

      DOUBLE PRECISION MM,MA,TF

      TF = (1/MM)*MA
```

```
      RETURN
      END

*******************************************************************
*   CALCULATE UNPERTURBED ORBIT
*******************************************************************

      SUBROUTINE UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,
     +   VI,VJ,VK,V,MU,PI,H,A,E,N,TA,P,MM,MA,EA,
     +   TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,TARAY,AINRAY,APRAY,TIMRAY,
     +   TT)
*     THIS SUBROUTINE CALCULATE THE UNPERTURBED ORBIT

*     THIS SUBROUTINE CALLS THE FOLLOWING SUBROUTINES:
*     NEWELT = CALCULATE NEW ELEMENTS AFTER TIME STEP
*     NEWPOS = CALCULATE NEW POSITION AFTER TIME STEP
*     NEWVEL = CALCULATE NEW VELOCITY AFTER TIME STEP
*     STORE  = STORES POSITION IN ARRAYS

      DOUBLE PRECISION T,DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                      MU,PI,H,A,E,N,TA,P,MM,MA,EA,TF,TT

      DIMENSION  RARAY(500),TARAY(500),RIRAY(500),RJRAY(500),
     +           RKRAY(500),AINRAY(500),APRAY(500),TIMRAY(500)

*     SET TRUE ANOMALY TO NEGATIVE SO LOOP CAN BE EXECUTED
      IF (TA .GT. 6.21) THEN
         TA = TA - (2*PI)
      ENDIF

*     CONTINUE AROUND ORBIT TILL CLOSE TO PERIGEE
 230  IF ((TA .LE. 6.21) .AND. (T .LE.PER)) THEN

*        Increment TRUE TIME
         TT = TT + DT
         CALL NEWELT(MM,MA,E,EA,TA,TF,DT,PI,PER)
         CALL NPOS(RI,RJ,RK,R,LAN,AP,I, TA,A,E)
         CALL NVEL(E,P,TA,LAN,AP,I,VI,VJ,VK,V,MU)

*        INCREMENT STEP COUNTER AND STORE VALUES
         NUM = NUM + 1
         CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,
     +              RARAY,TARAY,NUM,I,AP,AINRAY,APRAY,
     +              TT,TIMRAY)

*        INCREMENT TIME STEP COUNTER
         T= T + DT
         GOTO 230
      ENDIF
      RETURN
      END

*****************************************************************
* CALCULATE THE UNPERTURBED NEW ELEMENTS
*****************************************************************
```

```
      SUBROUTINE NEWELT(MM,MA,E,EA,TA,TF,DT,PI,PER)
*     THIS SUBROUTINE CALCULATES THE Unperturbed NEW ELEMENTS

*     THIS SUBROUTINE CALLS THE FOLLOWING SUBROUTINES:
*     NEA = NEW ECCENTRIC ANOMALY
*     NTA = NEW TRUE ANOMALY

      DOUBLE PRECISION MM,MA,E,EA,TA,TF,DT,PI,PER

*     Increment TIME OF FLIGHT AND CHECK IF TF GREATER THAN PERIOD
      TF = TF + DT
      IF (TF .GT. PER) THEN
         TF = TF - PER
      ENDIF

*     CALCULATE MEAN ANOMALY AND USE TO FIND ECCENTRIC Anomaly THEN NEW
*     TRUE ANOMALY
      MA = MM*(TF)
      CALL NEA(MA,E,EA)
      CALL NTA(EA,E,TA,PI)
      RETURN
      END

***********************************************************************
*  CALCULATE PERTURBED ORBIT
***********************************************************************

      SUBROUTINE PRETUR(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,
     +   VI,VJ,VK,V,FR,FS,FW,MU,PI,H,A,E,N,TA,P,MM,MA,EA,
     +   TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,TARAY,AINRAY,APRAY,TIMRAY,
     +   TT,TFEA,TFSU,TFMO,TFDRA,TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP)
*     THIS SUBROUTINE CALCULATES THE PERTURBED ORBIT.

*     THIS SUBROUTINE CALLS THE FOLLOWING SUBROUTINES:
*     TFORCE = CALCULATE THE TOTAL PERTURBING FORCE ON THE SATELLITE
*     PNEWEL = CALCULATE THE Perturbed NEW ELEMENTS
*     NPOS   = NEW POSITION AFTER TIME STEP
*     NVEL   = NEW VELOCITY AFTER TIME STEP
*     PERIOD = PERIOD OF PERTURBED ORBIT
*     STORE  = STORE POSITION AND ELEMENTS IN ARRAYS FOR PLOTTING

      DOUBLE PRECISION T,DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +             FR,FS,FW,MU,PI,H,A,E,N,TA,P,MM,MA,EA,TF,TT,
     +   DI,DA,DE,DMM,DMA,DLAN,DH,DAP,EI,EJ,EK,HI,HJ,LP,M,
     +   DVR,DVS,DVW,DVI,DVJ,DVK

      DIMENSION  RARAY(500),TARAY(500),RIRAY(500),RJRAY(500),
     +           RKRAY(500),AINRAY(500),APRAY(500),TIMRAY(500)

*     SET MEAN RADIUS OF EARTH
      RE = 6400.0

      DT = PER/50
      T = DT
      IF (TA .GT. 6.21) THEN
         TA = TA - (2*PI)
```

```
      ENDIF
      IF (TF .GE. PER) THEN
         TF = TF - PER
      ENDIF

*     CONTINUE Around ORBIT FOR ONE PERIOD
 240  IF ((TF .LT. PER) .AND. (T .LT. PER)) THEN

*        INCREMENT TRUE TIME
         TT = TT + DT
         CALL TFORCE(AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                TT,FR,FS,FW,MU,PI,
     +                FEA,FSU,FMO,FDRA,FOR,
     +               EI,EJ,EK,E,A,T,LP,TA,PER,EA,MA,TF,P,
     +               MM,N,H,HI,HJ,DT)
            CALL PNEWEL(FR,FS,FW,H,R,A,E,M,TA,DT,I,LAN,AL,
     +                     AP,P,MM,MA,EA,TF,T,MU,PI,
     +                  DI,DA,DE,DMM,DMA,DLAN,DH,DAP)
            CALL NPOS(RI,RJ,RK,R,LAN,AP,I, TA,A,E)
            CALL NVEL(E,P,TA,LAN,AP,I,VI,VJ,VK,V,MU)

*           CALCULATE NEW PERIOD AND RESET TIME STEP AND TIME COUNTER
*           IF NOT AT END OF ORBIT
            IF (T .LT. (PER-DT)) THEN
               CALL PERIOD(A,PER,PI,MU)
               DT = PER/50
               T = TF
            ENDIF

*        INCREMENT STEP COUNTER
         NUM = NUM + 1
 241     CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,
     +                       RARAY,TARAY,NUM,I,AP,AINRAY,APRAY,
     +                       TT,TIMRAY)

*        TOTAL ELEMENT CHANGES
         TDI = TDI + SNGL(ABS(DI))
         TDA = TDA + SNGL(ABS(DA))
         TDE = TDE + SNGL(ABS(DE))
         TDMM = TDMM + SNGL(ABS(DMM))
         TDMA = TDMA + SNGL(ABS(DMA))
         TDLAN = TDLAN + SNGL(ABS(DLAN))
         TDH = TDH + SNGL(ABS(DH)'
         TDAP = TDAP + SNGL(ABS(DAP))
         TFEA = TFEA + FEA
         TFSU = TFSU + FSU
         TFMO = TFMO + FMO
         TFDRA = TFDRA + FDRA

*        CHECK FOR IMPACT
         IF (R .LE. RE) THEN
              PRINT*,'SATELLITE WILL IMPACT THE EARTH!!'
              T = PER
         ENDIF

*        INCREMENT TIME COUNTER
```

```
      T = T + DT
      GOTO 240
   ENDIF
   RETURN
   END

**************************************************************************
* CALCULATE THE PERTURBING FORCES
**************************************************************************

      SUBROUTINE TFORCE(AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,TT,
     +                  FR,FS,FW,MU,PI,FEA,FSU,FMO,FDRA,FOR,
     +                  EI,EJ,EK,E,A,T,LP,TA,PER,EA,MA,TF,P,
     +                  MM,N,H,HI,HJ,DT)
*     THIS SUBROUTINE SUMS ALL THE PERTURBING FORCES FOR THE TOTAL
*     PERTURBING FORCE.

*     THE FOLLOWING SUBROUTINES WERE CALLED:
*     OBERT  = OBLATENESS OF THE EARTH
*     FSUN   = GRAVITATIONAL Attraction OF THE SUN
*     FMOON  = GRAVITATIONAL Attraction OF THE MOON
*     FDRAG  = DRAG FORCES

      DOUBLE PRECISION  FER,FES,FEW,FSR,FSS,FSW,FMR,FMS,FMW,MU,PI,
     +   FDR,FDS,FDW,FR,FS,FW,RI,RJ,RK,R,AL,I,TT,LAN,AP,VI,VJ,VK,V,
     +                  EI,EJ,EK,E,A,T,LP,TA,PER,EA,MA,TF,P,
     +                  MM,N,H,HI,HJ,DT

      CALL OBEART(RI,RJ,RK,R,AL,I,FER,FES,FEW,MU)
      CALL FSUN(TT,RI,RJ,RK,R,FSR,FSS,FSW,PI)
      CALL FMOON(TT,RI,RJ,RK,R,FMR,FMS,FMW,PI)
      CALL FDRAG(RI,RJ,RK,R,VI,VJ,VK,V,LAN,AP,I,FDR,FDS,FDW,
     +                  EI,EJ,EK,E,A,LP,TA,PER,EA,MA,AL,TF,P,PI,MU,
     +                  MM,N,H,HI,HJ,DT)

*     SUM VECTOR FORCES
      FR = FER + FSR + FMR + FDR
      FS = FES + FSS + FMS + FDS
      FW = FEW + FSW + FMW + FDW

*     CALCULATE TOTAL FORCE FROM EACH, AND TOTAL OF ALL
      FEA = SNGL(SQRT((FER**2)+(FES**2)+(FEW**2)))
      FSU = SNGL(SQRT((FSR**2)+(FSS**2)+(FSW**2)))
      FMO = SNGL(SQRT((FMR**2)+(FMS**2)+(FMW**2)))
      FDRA = SNGL(SQRT((FDR**2)+(FDS**2)+(FDW**2)))
      FOR = SNGL(SQRT((FR**2)+(FS**2)+(FW**2)))

      RETURN
      END

**************************************************************************

      SUBROUTINE OBEART(RI,RJ,RK,R,AL,I,FER,FES,FEW,MU)
*     THIS SUBROUTINE CALCULATES THE PERTURBING FORCE DUE TO THE
*     OBLIQUENESS OF THE EARTH.
```

```
      DOUBLE PRECISION  J2,RE,FER,FES,FEW,RI,RJ,RK,R,AL,I,MU,M

      J2 = 1.082364D-03
      RE = 6.3782D+03

      FER = ((-3.0D+00*MU*J2*(RE**2))/(2.0D+00*(R**4)))*
     +      (1.0D+00 - (3.0D+00* ((DSIN(I))**2) * ((DSIN(AL))**2)))
      FES = ((-3.0D+00*MU*J2*(RE**2))/(R**4))*
     +      (((DSIN(I))**2)* (DSIN(AL))*(DCOS(AL)))
      FEW = ((-3.0D+00*MU*J2*(RE**2))/(R**4))*
     +      (DSIN(I)*DCOS(I)*DSIN(AL))
      RETURN
      END

**********************************************************************

      SUBROUTINE FSUN(TT,RI,RJ,RK,R,FSR,FSS,FSW,PI)
*     THIS SUBROUTINE CALCULATES THE PERTURBING FORCE DUE TO THE SUN

*     THE FOLLOWING SUBROUTINES ARE CALLED:
*     SUNPOS = SUNS POSITION ORBITING AROUND EARTH
*     HEVBOD = PERTURBING FORCE FROM A Heavenly BODY

      DOUBLE PRECISION FSR,FSS,FSW,PI,
     +    RSI,RSJ,RSK,SLAN,SI,SAL,SMU,TT,RI,RJ,RK,R,RS

*     SUNS GRAVITATIONAL PARAMETER
      SMU = 1.32715440D+11
      CALL SUNPOS(TT,RSI,RSJ,RSK,RS,SLAN,SI,SAL,PI)
      CALL HEVBOD(RI,RJ,RK,R,RSI,RSJ,RSK,RS,SLAN,SAL,SI,SMU,FSR,FSS,FSW)
      RETURN
      END

**********************************************************************

      SUBROUTINE FMOON(TT,RI,RJ,RK,R,FMR,FMS,FMW,PI)
*     THIS SUBROUTINE CALCULATES THE PERTURBING FORCE DUE TO The MOON

*     THE FOLLOWING SUBROUTINE ARE CALLED:
*     MONPOS = MOONS POSITION ORBITING AROUND THE EARTH
*     HEVBOD = PERTURBING FORCE FROM A HEAVENLY BODY

      DOUBLE PRECISION  FMR,FMS,FMW,RMI,RMJ,RMK,MLAN,MI,MAL,MMU,
     +             TT,RI,RJ,RK,R,RM,PI

*     MOONS GRAVITATIONAL PARAMETER
      MMU = 4.90287D+03

      CALL MONPOS(TT,RMI,RMJ,RMK,RM,MLAN,MI,MAL,PI)
      CALL HEVBOD(RI,RJ,RK,R,RMI,RMJ,RMK,RM,MLAN,MAL,MI,MMU,FMR,FMS,FMW)
      RETURN
      END

**********************************************************************

      SUBROUTINE HEVBOD(RI,RJ,RK,R,RPI,RPJ,RPK,RP,LAN,AL,INC,MUP,
```

```
     +    FHR,FHS,FHW)
*     THIS SUBROUTINE CALCULATES THE PERTURBING FORCE DUE TO A
*     HEAVENLY BODY.

*     THE FOLLOWING SUBROUTINE WAS CALLED:
*     IJKRSW = 'IJK' SYSTEM TO THE 'RSW' SYSTEM

      DOUBLE PRECISION  DOT,FHI,FHJ,FHK,RI,RJ,RK,R,RPI,RPJ,RPK,RP,
     +    LAN,AL,INC,MUP,I,J,K,IP,JP,KP,M,FHR,FHS,FHW

*     CALCULATE UNIT VECTOR FOR SATELLITE AND PERTURBING BODIES POSITION
      I = RI/R
      J = RJ/R
      K = RK/R
      IP = RPI/RP
      JP = RPJ/RP
      KP = RPK/RP

*     CALCULATE DOT PRODUCT OF UNIT VECTORS
      DOT = (( I*IP )+( J*JP )+( K*KP ))

*     CALCULATE FORCES IN THE 'IJK' SYSTEM
      FHI = (MUP/(RP**2))*(R/RP)*(3.0D+00*DOT*(IP)-(I))
      FHJ = (MUP/(RP**2))*(R/RP)*(3.0D+00*DOT*(JP)-(J))
      FHK = (MUP/(RP**2))*(R/RP)*(3.0D+00*DOT*(KP)-(K))

*     Transform FORCES TO THE RSW SYSTEM
      CALL IJKRSW(LAN,AL,INC,FHI,FHJ,FHK,FHR,FHS,FHW)
      RETURN
      END

********************************************************************

      SUBROUTINE SUNPOS(TT,RSI,RSJ,RSK,RS,SLAN,SI,SAL,PI)
*     THIS SUBROUTINE CALCULATES THE SUNS POSITION

*     VARIABLES USED TO DESCRIBE THE SUNS ORBIT:
*     SI   = SUNS INCLINATION
*     SLAN= SUNS Longitude OF ASCENDING NODE
*     SAP  = SUNS ARGUMENT OF PERIGEE
*     RS   = SUNS ORBITAL RADIUS
*     STA  = SUNS TRUE ANOMALY
*     SAL  = SUNS ARGUMENT OF LONGITUDE

      DOUBLE PRECISION  SLAN,SI,SAL,RS,STA,SAP,TT,RSI,RSK,
     +    RSJ,RSP,RSQ,RSW,PI

      SI = 4.09279709D-01
      SLAN = 0.0D+00
      SAP = 0.0D+00
      RS = 1.4959965D+08
      STA = ((2.0*PI)/(365.0 * 86400.0) * TT)
      SAL = STA + SAP

*     CALCULATE SUNS POSITION IN 'PQW' SYSTEM
      RSP = RS*DCOS(STA)
```

```
      RSQ = RS*DSIN(STA)
      RSW = 0.0D+00

*     TRANSFORM POSITION TO 'IJK' SYSTEM
      CALL PQWIJK(SLAN,SAP,SI,RSP,RSQ,RSW,RSI,RSJ,RSK)
      RETURN
      END

***********************************************************************

      SUBROUTINE MONPOS(TT,RMI,RMJ,RMK,RM,MLAN,MI,MAL,PI)
*     THIS SUBROUTINE CALCULATES THE MOONS POSITION

*     VARIABLES USED TO DESCRIBE THE SUNS ORBIT:
*     MI   = MOONS INCLINATION
*     MLAN= MOONS Longitude OF ASCENDING NODE
*     MAP = MOONS ARGUMENT OF PERIGEE
*     RM   = MOONS ORBITAL RADIUS
*     MTA = MOONS TRUE ANOMALY
*     MAL = MOONS ARGUMENT OF LONGITUDE

      DOUBLE PRECISION    MLAN,MAL,RM,TM,MTA,RMP,RMQ,RMW,
     +    RMI,RMJ,RMK,[illegible],PI

      MI = 4.99164166D-01
      RM = 3.844D+05
      MLAN = 0.0
      MTA = ((2.0*PI)/(27.3 * 3600) * TT)
      MAP = 0.0D+00
      MAL = MTA

*     CALCULATE MOON POSITION IN 'PQW' SYSTEM
      RMP = RM*DCOS(MTA)
      RMQ = RM*DSIN(MTA)
      RMW = 0

*     TRANSFORM POSITION TO 'IJK' SYSTEM
      CALL PQWIJK(MLAN,MAP,MI,RMP,RMQ,RMW,RMI,RMJ,RMK)
      RETURN
      END

***********************************************************************

      SUBROUTINE FDRAG(RI,RJ,RK,R,VI,VJ,VK,V,LAN,AP,I,FDR,FDS,FDW,
     +                 EI,EJ,EK,E,A,T,LP,TA,PER,EA,MA,AL,TF,P,PI,MU,
     +                 MM,N,H,HI,HJ,DT)
*     THIS SUBROUTINE CALCULATES THE PERTURBING FORCE DUE TO DRAG

*     THE FOLLOWING VARIABLES ARE USED TO MODEL THE ATMOSPHERE:
*     RE       = RADIUS OF EARTH
*     M        = MASS OF SATELLITE
*     AR       = FRONTAL SURFACE AREA OF SATELLITE
*     Z        = ALTITUDE OF SATELLITE
*     K        = EXPONENTIAL DECAY FACTOR
*     DENO     = NORMAL DENSITY
*     CD       COEFFICIENT OF DRAG
```

```
      DOUBLE PRECISION  MAG,M,K,FDR,FDS,FDW,RE,AR,Z,DENO,CD,DEN,
     +                  FDJ,FDK,FDI,RI,RJ,RK,VI,VJ,VK,V,LAN,AP,I,R,
     +                  EI,EJ,EK,E,A,T,LP,TA,PER,EA,MA,AL,TF,P,PI,MU,
     +                  MM,N,H,HI,HJ,DT,DVR,DVS,DVW,DVI,DVJ,DVK

      RE = 6.378145D+03
      M = 1.0D+02
      AR = 2.0D+01
      Z = R - RE

*     DEPENDING ON ALTITUDE SET ATMOSPHERE VARIABLES
      IF (Z.LE.1.5D+02) THEN
         K = 4.74D-02
         DENO = 1.225D+00
         CD = 1.0D+00
      ELSEIF (Z.LE.5.5D+02) THEN
         K = 3.4614D-02
         DENO = 1.79846D-01
         CD = 2.0D+00
      ELSE
         K = 2.21698D-3
         DENO = 1.015484D-07
         CD = 2.0D+00
      ENDIF

*     CALCULATE ATMOSPHERIC DENSITY
      DEN = DENO * DEXP(-K*Z)

*     CALCULATE MAGNITUDE OF DRAG FORCE AND LIMIT IT TO 1.0E-20
      MAG = -(0.5D+00)*CD*AR*DEN*V*(1.0D-03)/M
      IF (ABS(MAG) .LT. 1.0D-20) THEN
         MAG = -1.0D-20
      ENDIF

*     GIVE DRAG FORCE A Direction OF MINUS THE VELOCITY
      FDR = 0.0
      FDS = MAG * V
      FDW = 0.0
      RETURN
      END

***********************************************************************
* CALCULATE PERTURBED NEW ELEMENTS
***********************************************************************

      SUBROUTINE PNEWEL(FR,FS,FW,H,R,A,E,N,TA,DT,I,LAN,AL,AP,P,
     +   MM,MA,EA,TF,T,MU,PI,DI,DA,DE,DMM,DMA,DLAN,DH,DAP)
*     THIS SUBROUTINE CALCULATES THE NEW ELEMENTS FROM THE PREVIOUS
*     ELEMENTS ADDED TO THE RATES OF CHANGE FOR ONE STEP

*     THE FOLLOWING SUBROUTINES ARE CALLED:
*     RATE   = CALCULATES RATES OF CHANGE OF ORBITAL ELEMENTS
*     NANGMO = NEW ANGULAR MOMENTUM (NEWH)
*     NSMA   = NEW SEMI-MAJOR AXIS (NEWA)
*     NECC   = NEW ECCENTRICITY (NEWE)
```

```
*     NINCL  = NEW INCLINATION (NEWI)
*     NASNOD = NEW LONGITUDE OF ASCENDING NODE (NEWLAN)
*     NARPER = NEW ARGUMENT OF PERIGEE ( NEWAP)
*     NMNMO  = NEW MEAN MOTION (NEWMM)
*     MEANMO = MEAN MOTION (MM)
*     NMMAN  = NEW MEAN ANOMALY (NEWMA)
*     NEA    = NEW ECCENTRIC ANOMALY (EA)
*     NTA    = NEW TRUE ANOMALY (TA)
*     TFLGHT = TIME OF FLIGHT (TF)

      DOUBLE PRECISION  FR,FS,FW,DMM,H,R,A,E,N,TA,DT,I,LAN,AL,AP,P,
     +   MM,MA,EA,TF,T,MU,PI,DA,DH,DE,DI,DLAN,DAP,DMA,
     +   NEWH,NEWA,NEWE,NEWI,NEWLAN,NEWAP,NEWMM

*     INCREMENT TIME OF FLIGHT BY ONE TIME STEP AND CALCULATE RATES
      TF = TF + DT
      CALL RATES(DH,DA,DE,DI,DLAN,DAP,DMM,DMA,E,MM,R,A,FR,FS,FW,
     +              TA,AL,H,P,T,MU,I)

*     CALCULATE NEW ELEMENTS
      CALL NANGMO(H,DT,DH,NEWH)
      CALL NSMA(A,DT,DA,NEWA)
      CALL NECC(E,DT,DE,NEWE)
      CALL NINCL(I,DT,DI,NEWI)
      CALL NASNOD(LAN,DT,DLAN,NEWLAN)
      CALL NARPER(AP,DT,DAP,NEWAP)

*     SET ELEMENTS TO NEW ELEMENTS
      A = NEWA
      E = NEWE
      I = NEWI
      LAN = NEWLAN
      AP = NEWAP
      P = A * (1 - E**2)

*     MOVE THE SATELLITE ONE TIME STEP
      CALL MEANMO(A,MM,MU)
      CALL NMNAN(MA,MM,DT,TF,DMA,PI)
      CALL NEA(MA,E,EA)
      CALL NTA(EA,E,TA,PI)
      CALL TFLGHT(MM,MA,TF)
      AL = TA + AP
      RETURN
      END

*********************************************************************
* CALCULATE THE RATES OF CHANGE OF THE ORBITAL Elements
*********************************************************************

      SUBROUTINE RATES(DH,DA,DE,DI,DLAN,DAP,DMM,DMA,E,MM,R,A,FR,FS,FW,
     +  TA,AL,H,P,T,MU,I)
*     THIS SUBROUTINE Calls THE FOLLOWING SUBROUTINES TO CALCULATE THE
*     TIME RATE-OF- CHANGE OF THE ORBITAL ELEMENTS:
*     RSMAX  = RATE-OF-CHANGE OF THE SEMI-MAJOR AXIS (DA)
*     RECC   = RATE-OF-CHANGE OF THE ECCENTRICITY (DE)
*     RINC   = RATE-OF-CHANGE OF THE INCLINATION (DI)
```

```
*     RLAN   = RATE-OF-CHANGE OF THE Longitude OF THE ASCENDING NODE
*              (DLAN)
*     RAP    = RATE-OF-CHANGE OF THE ARGUMENT OF PERIGEE (DAP)
*     RMM    = RATE-OF-Change OF THE MEAN MOTION (DMM)
*     RMA    = RATE-OF-CHANGE OF THE MEAN ANOMALY (DMA)
*     RANGMO = RATE-OF-CHANGE OF THE ANGULAR MOMENTUM (DH)

      DOUBLE PRECISION DH,DA,DE,DI,DLAN,DAP,DMM,DMA,E,MM,R,A,FR,FS,FW,
     +  TA,AL,H,P,T,MU,I

      CALL RSMAX(E,MM,R,A,FR,FS,DA,TA)
      CALL RECC(E,MM,R,A,FR,FS,TA,DE)
      CALL RINC(E,MM,R,A,FW,AL,DI)
      CALL RLAN(E,MM,R,A,I,FW,AL,DLAN)
      CALL RAP(E,MM,R,A,I,H,P,AL,TA,FR,FS,FW,DAP)
      CALL RMM(MM,A,DMM,DA,MU)
      CALL RMA(E,MM,R,A,TA,DMM,FR,FS,DMA,T)
      CALL RANGMO(R,FS,FW,DH)
      RETURN
      END

*************************************************************************

      SUBROUTINE RANGMO(R,FS,FW,DH)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE
*     ANGULAR MOMENTUM

      DOUBLE PRECISION  FS,FW,DHW,DHS,DH,R

      DHW = R * FS
      DHS = R * FW
      DH = DSQRT((DHW**2) + (DHS**2))
      RETURN
      END

*************************************************************************

      SUBROUTINE RSMAX(E,MM,R,A,FR,FS,DA,TA)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE SEMI-MAJOR
*     AXIS

      DOUBLE PRECISION  DA,FR,FS,E,MM,R,A,TA,ET

*     TRAP (E) SO DENOMINATOR DOES NOT GOTO ZERO
      IF (E.GT.0.9) THEN
         ET = 0.9
      ELSE
         ET = E
      ENDIF
       DA = ((2.0D+00*E *DSIN(TA))/(MM *DSQRT(1.0D+00-(ET**2))))*FR +
     +    ((2.0D+00*A*DSQRT(1.0D+00-(E **2)))/(MM *R))*FS
      RETURN
      END

*************************************************************************
```

```
      SUBROUTINE RECC(E,MM,R,A,FR,FS,TA,DE)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE ECCENTRICITY

      DOUBLE PRECISION  DE,FR,FS,E,MM,R,A,TA,ET

*     TRAP (E) SO DENOMINATOR DOES NOT GOTO ZERO
      IF (E.LT.0.1) THEN
         ET = 0.1
      ELSE
         ET = E
      ENDIF
       DE = ((DSQRT(1.0D+00 - (E **2))*SIN(TA))/(MM *A))*FR +
     +   ((DSQRT(1.0D+00 - (E **2)))/(MM *ET*(A**2)))*
     +   ((A**2)*(1.0D+00 - (E **2))/(R) - (R))*FS
      RETURN
      END

*****************************************************************

      SUBROUTINE RLAN(E,MM,R,A,I,FW,AL,DLAN)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE LONGITUDE
*     OF THE ASCENDING NODE

      DOUBLE PRECISION  DLAN,FW,E,MM,R,A,I,AL,ET,IT

*     TRAP (E) AND (I) SO DENOMINATOR DOES NOT GOTO ZERO
      IF (E.GT.0.9) THEN
         ET = 0.9
      ELSE
         ET = E
      ENDIF
      IF (I.LT.0.01745) THEN
         IT = 0.01745
      ELSE
         IT = I
      ENDIF
      DLAN = (R*FW*DSIN(AL))/(MM *(A**2)*DSQRT(1.0D+00 - (ET**2))*
     +   DSIN(IT))
      RETURN
      END

*****************************************************************

      SUBROUTINE RAP(E,MM,R,A,I,H,P,AL,TA,FR,FS,FW,DAP)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE ARGUMENT
*     OF PERIGEE

      DOUBLE PRECISION  DAPR,DAPS,DAPW,DAP,FR,FS,FW,E,MM,R,I,H,P,AL,TA,
     +   ET,A,IT

*     TRAP (E) AND (I) SO DENOMINATOR DOES NOT GOTO ZERO
      IF (I.LT.0.01745) THEN
         IT = 0.01745
      ELSE
         IT = I
      ENDIF
```

```
      IF (E.GT.0.9) THEN
         ET = 0.9
      ELSEIF (E.LT.0.1) THEN
         ET = 0.1
      ELSE
         ET = E
      ENDIF
      DAPR = (-DSQRT(1.0+00 - (E **2))*DCOS(TA))/(MM *A*ET) * FR
      DAPS = (P/(ET*H))*(DSIN(TA))*
     +     (1.0D+00 + 1.0D+00/(1.0D+00 + ET*DCOS(TA))) *FS
      DAPW = (-R*(1.0D+00/DTAN(IT))*DSIN(AL))/
     +        (MM *(A**2)*DSQRT(1.0D+00 - (ET**2)))*FW
      DAP = DAPR + DAPS + DAPW
      RETURN
      END

***********************************************************************

      SUBROUTINE RINC(E,MM,R,A,FW,AL,DI)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE INCLINATION

      DOUBLE PRECISION  DI,FW,E,MM,R,A,AL,ET

*     TRAP (E) SO DENOMINATOR DOES NOT GOTO ZERO
      IF (E.GT.0.9) THEN
         ET = 0.9
      ELSE
         ET = E
      ENDIF
      DI = (R*FW*DCOS(AL))/(MM *(A**2)*DSQRT(1.0D+00 - (ET**2)))
      RETURN
      END

***********************************************************************

      SUBROUTINE RMM(MM,A,DMM,DA,MU)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE MEAN MOTION

      DOUBLE PRECISION  DMM,DA,MM,A,MU

      DMM =((-3.0D+00*MU)/(2.0D+00*MM *(A**4)))* DA
      RETURN
      END

***********************************************************************

      SUBROUTINE RMA(E,MM,R,A,TA,DMM,FR,FS,DMA,T)
*     THIS SUBROUTINE CALCULATES THE RATE OF CHANGE OF THE MEAN Anomaly

      DOUBLE PRECISION  DMAA,DMAB,DMAC,DMAD,DMM,FR,FS,DMA,E,MM,R,A,TA,
     +   ET,T

*     TRAP (E) SO DENOMINATOR DOES NOT GOTO ZERO
      IF (E.GT.0.9) THEN
         ET = 0.9
      ELSEIF (E.LT.0.1) THEN
```

```
         ET = 0.1
      ELSE
         ET = E
      ENDIF
      DMA  = (-1.0D+00/(MM *A))*
     +       (((2.0D+00*R)/A) - ((1 - (E **2))/ET)*DCOS(TA)) * FR -
     +      (1-(E **2))/(MM *A*ET)*(1+ R/(A*(1-(E**2))))*(SIN(TA)*FS)-
     +       (T * DMM)
      RETURN
      END

*****************************************************************
* CALCULATE THE NEW ORBITAL ELEMENTS
*****************************************************************

      SUBROUTINE NSMA(A,DT,DA,NEWA)
*     THIS SUBROUTINE CALCULATES THE NEW SEMI-MAJOR AXIS

      DOUBLE PRECISION DA,DT,A,NEWA

      NEWA = A + DA*DT
      RETURN
      END

*****************************************************************

      SUBROUTINE NECC(E,DT,DE,NEWE)
*     THIS SUBROUTINE CALCULATES THE NEW ECCENTRICITY

      DOUBLE PRECISION DE,DT,E,NEWE

      NEWE = E + DE*DT
      RETURN
      END

*****************************************************************

      SUBROUTINE NINCL(I,DT,DI,NEWI)
*     THIS SUBROUTINE CALCULATES THE NEW INCLINATION

      DOUBLE PRECISION DI,DT,I,NEWI

      NEWI = I + DI*DT
      RETURN
      END

*****************************************************************

      SUBROUTINE NASNOD(LAN,DT,DLAN,NEWLAN)
*     THIS SUBROUTINE CALCULATES THE NEW LONGITUDE OF THE ASCENDING NODE

      DOUBLE PRECISION DLAN,DT,LAN,NEWLAN

      NEWLAN = LAN + DLAN*DT
      RETURN
      END
```

```
************************************************************************

      SUBROUTINE NARPER(AP,DT,DAP,NEWAP)
*     THIS SUBROUTINE CALCULATES THE NEW ARGUMENT OF PERIGEE

      DOUBLE PRECISION DAP,DT,AP,NEWAP

      NEWAP = AP + DAP*DT
      RETURN
      END

************************************************************************

      SUBROUTINE NMNAN(MA,MM,DT,TF,DMA,PI)
*     THIS SUBROUTINE CALCULATES THE NEW MEAN Anomaly

      DOUBLE PRECISION  DMM,FR,FS,DMA,DT,MA,E,R,A,TA,MM,TF,T,PI

      MA = MM*(TF) + DMA*DT
      IF (MA .GT. (2*PI)) THEN
         MA = MA - (2*PI)
      ENDIF
      RETURN
      END

************************************************************************

      SUBROUTINE NMNMO(MM,DMM,DT,NEWMM)
*     THIS SUBROUTINE CALCULATE THE NEW MEAN MOTION

      DOUBLE PRECISION  DMM,DT,MM,NEWMM

      NEWMM = MM + DMM*DT
      RETURN
      END

************************************************************************

      SUBROUTINE NEA(MA,E,EA)
*     THIS SUBROUTINE CALCULATES THE NEW ECCENTRIC ANOMOLY BY USING
*     NEWTONS METHOD OF ROOT FINDING

      DOUBLE PRECISION  EAN,MAN,MA,E,EA,DIFF

*     LET (EA) EQUAL (MA) FOR INITIAL GUESS AT ROOT
      EA = MA
      EAN = EA + (MA - EA +E*DSIN(EA))/(1.0D+00 - E*DCOS(EA))
      MAN = EAN - E*SIN(EAN)

*     CHECK DIFFERENCE (DIFF)
      DIFF = ABS(MA -MAN)
      EA = EAN

*     CONTINUE TO INTERATE UNTIL DIFFERENCE IS NEGLIGIBLE
 200  IF(DIFF.GT.0.0000000001) THEN
```

```
      EAN = EA + (MA - EA + E*DSIN(EA))/(1.0D+00 - E*DCOS(EA))
      MAN = EAN - E*DSIN(EAN)
      EA = EAN
      DIFF = ABS(MA - MAN)
      GOTO 200
   ENDIF
   EA = EAN
   RETURN
   END

******************************************************************

   SUBROUTINE NTA(EA,E,TA,PI)
*  THIS SUBROUTINE CALCULATES THE NEW TRUE Anomaly

   DOUBLE PRECISION EA,E,TA,PI

   TA = DACOS((E - DCOS(EA))/(E*DCOS(EA) - 1.0D+00))
   IF (EA.GT.PI) THEN
      TA = (2*PI) - TA
   ENDIF
   RETURN
   END

*********************************************************************

   SUBROUTINE NANGMO(H,DT,DH,NEWH)
*  THIS SUBROUTINE CALCULATES THE NEW ANGULAR MOMENTUM

   DOUBLE PRECISION DH,DT,H,NEWH

   NEWH = H + DH*DT
   RETURN
   END

******************************************************************

   SUBROUTINE INTSUM(TFEA,TFSU,TFMO,TFDRA,TDI,TDA,TDE,TDMM,TDMA,
  +                  TDLAN,TDH,TDAP)
*  THIS SUBROUTINE INITIALIZES THE SUMS OF FORCES AND ELEMENT CHANGES

      TFEA = 0.0
      TFSU = 0.0
      TFMO = 0.0
      TFDRA = 0.0
      TDI = 0.0
      TDA = 0.0
      TDE = 0.0
      TDMM = 0.0
      TDMA = 0.0
      TDLAN = 0.0
      TDH = 0.0
      TDAP = 0.0
   RETURN
   END
```

```
*************************************************************************
* CALCULATE THE NEW POSITION AND VELOCITY VECTORS
*************************************************************************

      SUBROUTINE NPOS(RI,RJ,RK,R,LAN,AP,INC, TA,A,E)
*     THIS SUBROUTINE CALCULATES THE NEW POSITION VECTOR

      DOUBLE PRECISION XW,YW,ZW,INC,RI,RJ,RK,R,LAN,AP,TA,A,E

*     CALCULATE POSITION VECTOR IN 'PQW' SYSTEM
      R = (A*(1 - (E**2)))/(1 + E*DCOS(TA))
      XW = R*DCOS(TA)
      YW = R*DSIN(TA)
      ZW = 0

*     TRANSFORM POSITION TO 'IJK' SYSTEM
      CALL PQWIJK(LAN,AP,INC,XW,YW,ZW,RI,RJ,RK)
      R = DSQRT((RI**2) + (RJ**2) + (RK**2))
      RETURN
      END

************************************************************************

      SUBROUTINE NVEL(E,P,TA,LAN,AP,INC,VI,VJ,VK,V,MU)
*     THIS SUBROUTINE CALCULATES THE NEW VELOCITY VECTOR

      DOUBLE PRECISION  INC,VP,VQ,VW,MU,E,P,TA,LAN,AP,VI,VJ,VK,V

*     CALCULATE VELOCITY IN 'PQW' SYSTEM
      VP = DSQRT(MU/P)*(-DSIN(TA))
      VQ = DSQRT(MU/P)*(E + DCOS(TA))
      VW = 0.0D+00

*     TRANSFORM VELOCITY INTO 'IJK' SYSTEM
      CALL PQWIJK(LAN,AP,INC,VP,VQ,VW,VI,VJ,VK)
      V = DSQRT((VI**2) + (VJ**2) +(VK**2))
      RETURN
      END

****************************************************************************
*   VELOCITY CHANGE
****************************************************************************

      SUBROUTINE CHGVEL(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,
     +   VI,VJ,VK,V,MU,PI,H,A,E,N,TA,P,MM,MA,EA,
     +   TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,TARAY,AINRAY,APRAY,TIMRAY,
     +   TT,EI,EJ,EK,LP,HI,HJ,IOPT1,TFEA,TFSU,TFMO,TFDRA,
     +     TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP)
*     THIS SUBROUTINE CALCULATE VELOCITY CHANGES

* THE FOLLOWING SUBROUTINES ARE CALLED:
*     TACHG  = RETURNS TRUE ANOMALY FOR VELOCITY CHANGE LOCATION (CHTA)
*                AND AN INDICATOR OF LOCATION (ITA)
*     CALCEL = CALCULATE Orbital ELEMENTS
*     UNPRET = CALCULATE UNPERTURBED ORBIT
```

```
*      NPOS    = CALCULATE NEW POSITION
*      NVEL    = CALCULATE NEW VELOCITY
*      STORE   = STORE POSITION AND ELEMENTS IN ARRAYS
*      ENERGY = ENERGY OF SATELLITE
*      ECC     = ECCENTRICITY
*      SMAXIS = SEMI-MAJOR AXIS

       DOUBLE PRECISION T,DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                        MU,PI,H,A,E,N,TA,P,MM,MA,EA,TF,TT,
     +   NEWVI,NEWVJ,NEWVK,NEWV,VMAX,CHTA,EI,EJ,EK,LP,HI,HJ,VCIR,
     +   DI,DE,DA,DMM,DMA,DLAN,DH,DAP,NAWEI,NEWEJ,NEWEK,NEWE,NEWENR,
     +   NEWA,NEWRP,RE

       DIMENSION  RARAY(500),TARAY(500),RIRAY(500),RJRAY(500),
     +            RKRAY(500),AINRAY(500),APRAY(500),TIMRAY(500)

       CHARACTER*1,YORN,PYORN

       RE = 6.3782D+03

*      PROMPT THE USER FOR THE VELOCITY Change LOCATION
       CALL TACNG(PI,CHTA,ITA)

*      SET TIME COUNTER TO ONE TIME STEP
       T = DT

*      ROTATE TO THE VELOCITY CHANGE LOCATION
*      THIS IS IDENTICAL TO THE Unperturbed ORBIT WITH THE EXCEPTION
*      THAT A COMPLETE ORBIT IS NOT CALCULATED
       PRINT*,'ROTATE TO VELOCITY CHANGE LOCATION'
       IF ((ITA.EQ.2) .OR. (ITA.EQ.3)) THEN
          PRINT*,'BEFORE TA =',TA
          IF (TA .GT. 6.21) THEN
             TA = TA - (2*PI)
          ENDIF
 250      IF((T.LE.PER).AND.(TA.LT.CHTA)) THEN
*              PRINT*,'TA =',TA
               NUM = NUM + 1
               TT = TT + DT
                    CALL NEWELT(MM,MA,E,EA,TA,TF,DT,PI,PER)
               CALL NPOS(RI,RJ,RK,R,LAN,AP,I, TA,A,E)
               CALL NVEL(E,P,TA,LAN,AP,I,VI,VJ,VK,V,MU)
               CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,RARAY,
     +                   TARAY,NUM,I,AP,AINRAY,APRAY,TT,TIMRAY)
               T = T + DT
               GOTO 250
          ENDIF
          IF (TF .GE. PER) THEN
               TF = TF - PER
          ENDIF
       ENDIF

*      PRINT ESCAPE VELOCITY AND CIRCULAR VELOCITY FOR Reference
       CALL EXCMS('CLRSCRN')
       PRINT*,'AFTER TA =',TA
       PRINT*,'THIS SHOULD BE THE DESIRED RADIUS RP OR RA'
```

```
 260  PRINT*,'RADIUS =',R
      PRINT*,'VELOCITY =',V
      VMAX = DSQRT(2.0*(MU / R))
      PRINT*,'MAX VELCCITY AT THIS RADIUS IS:',VMAX
      VCIR = DSQRT(MU/R)
      PRINT*,'CIRCULAR VELOCITY AT THIS RADIUS IS :',VCIR

*     PROMPT USER TO CHANGE VELOCITY IN ORBITAL PLANE
      PRINT*,'DO YOU WANT TO CHANGE THE VELOCITY IN THE ORBITAL PLANE?'
      PRINT*,'ENTER "Y" OR "N" :'
      READ*,PYORN
      PRINT*,PYORN
      IF (PYORN .EQ. 'Y') THEN
         PRINT*,'GIVE THE TOTAL CHANGE IN VELOCITY, I.E. 5.0 KM.'
         PRINT*,'THE PROGRAM WILL FIGURE OUT THE FINAL VELOCITY VECTOR'
         PRINT*,'  ENTER VELOCITY CHANGE:'
         READ*,CHGV
         PRINT*,CHGV

*        CALCULATE NEW VELOCITY FOR CHANGE IN THE ORBITAL PLANE
         NEWVI = VI + (CHGV * VI / V)
         NEWVJ = VJ + (CHGV * VJ / V)
         NEWVK = VK + (CHGV * VK / V)

*     Velocity CHANGE OUT OF ORBITAL PLANE
      ELSEIF (PYORN .EQ. 'N') THEN
         PRINT*,' ENTER THE NEW VELOCITY VECTOR:'
         PRINT*,' ENTER THE NEW VI'
         READ*,NEWVI
         PRINT*,NEWVI
         PRINT*,' ENTER THE NEW VJ'
         READ*,NEWVJ
         PRINT*,NEWVJ
         PRINT*,' ENTER THE NEW VK'
         READ*,NEWVK
         PRINT*,NEWVK
         NUM = 1
         ITA = 3
      ELSE
         CALL EXCMS('GLRSCRN')
         GOTO 260
      ENDIF

*     PRINT NEW VELOCITY FOR USER TO CHECK
      NEWV = DSQRT((NEWVI**2) + (NEWVJ**2) + (NEWVK**2))
      PRINT*,'NEW VI =',NEWVI
      PRINT*,'NEW VJ =',NEWVJ
      PRINT*,'NEW VK =',NEWVK
      PRINT*,'NEW V  =',NEWV
      PRINT*,' ARE THESE VALUES THE ONES YOU WANT?'
      PRINT*,'ENTER "Y" OR "N" :'
      READ*,YORN
      PRINT*,YORN
      IF (YORN  .EQ. 'N') THEN
         CALL EXCMS('CLRSCRN')
         GOTO 260
```

```
      ENDIF

*     CHECK FOR VALID VELOCITY
      IF ( NEWV .GT. VMAX) THEN
         PRINT*,'YOUR VELOCITY IS TO GREAT !!'
         GOTO 260
      ENDIF

*     Calculate PERIGEE RADIUS TO SEE IF SATELLITE WILL IMPACT EARTH
      CALL ENERGY(NEWV,R,MU,NEWENR)
      CALL ECC(RI,RJ,RK,R,NEWVI,NEWVJ,NEWVK,NEWV,NEWEI,NEWEJ,NEWEK,
     +         NEWE,MU)
      CALL SMAXIS(MU,NEWENR,NEWA)
      NEWRP = NEWA*(1.0 - NEWE)
      IF (NEWRP .LE. RE) THEN
         PRINT*,'YOUR VELOCITY AT THIS POINT IS TO SMALL!!!'
         PRINT*,'THE SATELLITE WILL IMPACT THE EARTH!!'
         PRINT*,'THE SATELLITES RADIUS OF PERIGEE WOULD BE ',NEWRP
         PRINT*,'A NEW VELOCITY WILL HAVE TO BE ENTERED!!'
         GOTO 260
      ENDIF

*     ACCEPT NEW VELOCITY
      VI = NEWVI
      VJ = NEWVJ
      VK = NEWVK
      V = NEWV

*     CALCULATE NEW ELEMENT WITH NEW VELOCITY AND SET TIME STEP
      CALL CALCEL( RI,RJ,RK,R,VI,VJ,VK,V,EI,EJ,EK,E,A,I,LAN,LP,TA,
     +            PER,EA,MA,AP,AL,TF,P,PI,MU,MM,N,H,HI,HJ)
      DT = PER/50.0
      T = DT

*     THE FOUR Different CASES OF VELOCITY CHANGES FOLLOWS:

*     VELOCITY CHANGE AT PERIGEE, AND NEWV > V Circular
      IF((ITA.EQ.1).AND.(NEWV.GT.VCIR))THEN
         CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                             MU,PI,H,A,E,N,TA,P,MM,
     +                 MA,EA,TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +                 TARAY,AINRAY,APRAY,TIMRAY,TT)

*     Change VELOCITY AT PERIGEE, AND NEWV <= V CIRCULAR
*     APOGEE AND PERIGEE SWAP
      ELSEIF ((ITA.EQ.1).AND.(NEWV.LE.VCIR))THEN

*        CLEAR PREVIOUS PLOTS
         NUM = 1
         CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,RARAY,TARAY,
     +          NUM,I,AP,AINRAY,APRAY,TT,TIMRAY)
         T = PER/2

*        STEP SATELLITE TO NEW PERIGEE, ONLY A HALF ORBIT
         CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                             MU,PI,H,A,E,N,TA,P,MM,
```

```
     +                 MA,EA,TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +                 TARAY,AINRAY,APRAY,TIMRAY,TT)

*        RESET TIME COUNTER TO ONE TIME STEP
         T = DT

*        CALCULATE COMPLETE NEXT ORBIT
         CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                              MU,PI,H,A,E,N,TA,P,MM,
     +                 MA,EA,TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +                 TARAY,AINRAY,APRAY,TIMRAY,TT)

*     CHANGE VELOCITY AT APOGEE, AND NEW V < V CIRCULAR
      ELSEIF ((ITA.EQ.2) .AND.(NEWV .LT. VCIR)) THEN

         T = PER/2

*        FINISH ORBIT
         CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                              MU,PI,H,A,E,N,TA,P,MM,
     +                 MA,EA,TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +                 TARAY,AINRAY,APRAY,TIMRAY,TT)

*     CHANGE VELOCITY AT Apogee, AND NEWV >= V CIRCULAR
*     OR AT ANY OTHER TRUE Anomaly
      ELSEIF (((ITA.EQ.2).AND.(NEWV.GE.VCIR)) .OR. (ITA.EQ.3)) THEN

         IF (TA .GT. 6.21) THEN
            TA = TA - (2*PI)
         ENDIF

*        CLEAR PREVIOUS ORBITS AND STEP SATELLITE TO NEW PERIGEE
         T = TF
         NUM = 1
         CALL STORE(RI,RJ,RK,R,TA,RIRAY,RJRAY,RKRAY,RARAY,TARAY,
     +           NUM,I,AP,AINRAY,APRAY,TT,TIMRAY)
         CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                              MU,PI,H,A,E,N,TA,P,MM,
     +                 MA,EA,TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +                 TARAY,AINRAY,APRAY,TIMRAY,TT)
         IF (TF .GE. PER) THEN
            TF = TF - PER
         ENDIF

*        CALCULATE COMPLETE NEXT ORBIT
         T = DT
         CALL UNPRET(DT,PER,AL,LAN,AP,I,RI,RJ,RK,R,VI,VJ,VK,V,
     +                              MU,PI,H,A,E,N,TA,P,MM,
     +                 MA,EA,TF,T,NUM,RIRAY,RJRAY,RKRAY,RARAY,
     +                 TARAY,AINRAY,APRAY,TIMRAY,TT)
      ENDIF
      RETURN
      END

**********************************************************************
```

```
      SUBROUTINE TACNG(PI,CHTA,ITA)
*     THIS SUBROUTINE Asks THE USER FOR VELOCITY CHANGE LOCATION

      DOUBLE PRECISION CHTA,PI

      CALL EXCMS('CLRSCRN')
      PRINT*,'WHERE DO YOU WANT TO CHANGE THE VELOCITY?'
      PRINT*,'    1. AT CURRENT PERIGEE'
      PRINT*,'    2. AT CURRENT Apogee'
      PRINT*,'    3. AT A SPECIFIC TRUE Anomaly'
      PRINT*,'ENTER "1", "2" OR "3"'
      READ*,ITA
      PRINT*,ITA

*     SET TRUE ANOMALY CHANGE LOCATION (CHTA) TO DESIRED LOCATION
      IF (ITA .EQ. 1) THEN
         CHTA = 0.0
      ENDIF
      IF (ITA .EQ. 2) THEN
         CHTA = PI
      ENDIF
      IF (ITA .EQ. 3) THEN
         PRINT*,'AT WHAT TRUE ANOMALY DO YOU WANT TO CHANGE THE'
         PRINT*,'VELOCITY?'
         PRINT*,'ENTER TRUE ANOMALY IN DEGREES'
         READ*,CHTA
         PRINT*,CHTA
         CHTA = CHTA * PI / 180
      ENDIF
      RETURN
      END

*****************************************************************************
*    OUTPUT PLOTS
*****************************************************************************

      SUBROUTINE PLOTS(RIRAY,RJRAY,RKRAY,RARAY,TARAY,NUM,PI,INC,LP,A,
     +                 E,TF,AINRAY,APRAY,TIMRAY,TFEA,TFSU,TFMO,TFDRA,
     +                 PER,TDI,TDA,TDE,TDMM,TDMA,TDLAN,TDH,TDAP,
     +                 MM,MA,LAN,H,AP,R,V)

*     THIS SUBROUTINE ASKS THE USER FOR THE TYPE OF OUTPUT THAT IS
*     DESIRED PERIFOCAL, GROUND TRACK OR TO SKIP THE PLOT.

*     THE FOLLOWING SUBROUTINES ARE CALLED:
*     PERIF  = PLOT PERIFOCAL ORBIT
*     GRTRK  = PLOT GROUND TRACK
*     DATE   = DISPLAYS DATA ON PLOT
*     TEC618 = SET Disspla TO TEC 618 OUTPUT
*     ENDPL  = END THIS DISSPLA PLOT
*     REFER TO DISSPLA USER"S MANUAL FOR EXPLANATION OF DISSPLA
*     SUBROUTINES

      DOUBLE PRECISION PI,A,E,INC,LP,TF,PER,MM,MA,LAN,H,AP,R,V

      DIMENSION RIRAY(500),RJRAY(500),RKRAY(500),RARAY(500),TARAY(500),
```

```
     +     AINRAY(500),APRAY(500),TIMRAY(500)
      CHARACTER*1,YORN

      CALL EXCMS('CLRSCRN')

* CALCULATE SINGLE PRECISION VARIABLES
      SPI = SNGL(PI)
      SA = SNGL(A)
      SE = SNGL(E)
      SINC = SNGL(INC)
      SLP = SNGL(LP)
      STF = SNGL(TF)
      SPER = SNGL(PER)
      SMM = SNGL(MM)
      SMA = SNGL(MA)
      SLAN = SNGL(LAN)
      SH = SNGL(H)
      SAP = SNGL(AP)
      SV = SNGL(V)
      SR = SNGL(R)

*     PROMPT USER FOR DISPLAY TYPE
 340  PRINT*,'WHAT TYPE OF Display IS DESIRED:'
      PRINT*,'     1. PERIFOCAL'
      PRINT*,'     2. GROUND TRACK'
      PRINT*,'     3. SKIP PLOT'
      PRINT*,'ENTER 1,2,3,4:'
      READ*,INPUT
      PRINT350,INPUT
 350  FORMAT(I4)

      CALL TEK618

*     CALL APPROPRIATE PLOT
      IF (INPUT .EQ. 1) THEN
         CALL PERIF(RARAY,TARAY,NUM,SPI,SINC,SLP,SA,SE)
      ELSEIF (INPUT .EQ. 2) THEN
         CALL GRTRK(AINRAY,APRAY,TARAY,STF,NUM,TIMRAY)
      ELSEIF (INPUT .EQ. 3) THEN
         GOTO 360
      ELSE
         PRINT*,'INVALID ENTRY!'
         GOTO 340
      ENDIF

*     DISPLAY DATA
      CALL DATA(SINC,SA,SE,TFEA,TFSU,TFMO,TFDRA,SPER,SPI,TDI,TDA,TDE,
     +          TDMM,TDMA,TDLAN,TDH,TDAP,SMM,SMA,SLAN,SH,SAP,SV,SR)
      CALL ENDPL(0)

*     PROMPT USER IF ANOTHER DISPLAY TYPE IS DESIRED
      PRINT*,'WOULD YOU LIKE ANOTHER PLOT USING THE SAME ORBITAL'
      PRINT*,'PARAMETERS AND DATA:'
      PRINT*,'ENTER "Y" OR "N" :'
      READ*,YORN
      PRINT*,YORN
```

```
      IF (YORN .EQ. 'Y') THEN
         GOTO 340
      ENDIF
 360  RETURN
      END

************************************************************************

      SUBROUTINE PERIF(RARAY,TARAY,NUM,PI,INC,LP,A,E)
*     THIS SUBROUTINE PLOTS OUT THE RESULTS OF THE PROGRAM USING THE
*     DISPLAY FEATURE ON THE MAIN FRAME.
*     REFER TO DISSPLA USERS GUIDE FOR EXPLANATION OF DISSPLA
*     SUBROUTINES.

      REAL INC,LP
      DIMENSION TARAY(500),RARAY(500),RIRAY(500),RJRAY(500),RKRAY(500)

      I = 1

*     SET SCALE OF AXIS
      RSTEP = (A*(1+E)) / 3
      CALL TEK618
      CALL RESET(3HALL)
      CALL SCMPLX
      CALL PHYSOR(1.25,4.)
      CALL AREA2D(6.,6.)
      CALL MESSAG('PERIFOCAL COORDINATE SYSTEM$',100,1.0,6.5)
      CALL XNAME('XW',2)
      CALL YNAME('YW',2)
      CALL XAXANG(90.0)
      CALL YAXANG(0.0)
      CALL INTAXS
      CALL POLAR(1.,RSTEP,3.,3.)
      CALL POLY3
      CALL NOCHEK
      CALL CURVE(TARAY,RARAY,NUM,1)
      CALL COMPLX
      CALL HEIGHT(.2)
      CALL RESET('COMPLEX')
      CALL RESET('HEIGHT')
      CALL ENDGR(0)

*     Display  EARTH PLOT
      CALL EARTH1(A,E,INC,PI,RSTEP)
      RETURN
      END

************************************************************

      SUBROUTINE EARTH1(A,E,INC,PI,RSTEP)
*     THIS SUBROUTINE PLOTS A VIEW OF THE WORLD, LOOKING DOWN THE 'Z'
*     AXIS, PLACED ON THE ORIGIN.  THE Latitude IS FIXED, BUT THE
*     LONGITUDE VARIES WITH THE INCLINATION.
*     REFER TO DISSPLA USER"S MANUAL FOR EXPLANATION OF DISSPLA
*     SUBROUTINES

```

```
      REAL INC
      COMMON IWORK(3800)
      DATA IWDIM/3800/

      RE = 6378.145

*     SCALE THE EARTH PLOT AND CENTER ON THE ORIGIN
      SCFAC = RE/RSTEP
      SCFAC2 = SCFAC * 2.0
      XPHS = 1.25 + 3.0 -  SCFAC
      YPHS = 4.0 + 3.0 -  SCFAC
      YPOLE = 90 - (INC * 180 / PI)
      IF(YPOLE .GT. 90) THEN
         YPOLE = YPOLE - 90
      ENDIF
      YORIG = YPOLE - 90
      YMAX = YPOLE + 90
      CALL RESET(3HALL)
      CALL PHYSOR(XPHS,YPHS)
      CALL PROJCT('LAMBERT EQ/AREA')
      CALL MAPOLE(0.0,YPOLE)
      CALL AREA2D (SCFAC2,SCFAC2)
      CALL THKFRM (0.02)
      CALL GRAF(-90.,30.,90.,YORIG,30.,YMAX)
      CALL FRAME
      CALL MAPFIL('MAPDTA')
      CALL LBLANK('LAND',IWDIM)
      CALL GRID(1,1)
      CALL LBLANK('WATER',IWDIM)
      CALL DASH
      CALL GRID(1,1)
      CALL RESET ('DASH')
      CALL ENDGR(0)
      RETURN
      END

*********************************************************************

      SUBROUTINE GRTRK(AINRAY,APRAY,TARAY,TF,NUM,TIMRAY)

      DIMENSION AINRAY(500),APRAY(500),TARAY(500),
     +   ELARAY(500),ELORAY(500),TLONG(500),TLAT(500),TIMRAY(500)

      RE = 6.3782E+03
      EROT = 7.292115856E-05
      STF = (TF)
      I = 1

*     LOAD ARRAYS WITH LATITUDE AND LONGITUDE
 410  IF (I .LE. NUM) THEN
         X = RE*COS(APRAY(I))*COS(TARAY(I))-RE*SIN(APRAY(I))*
     +         SIN(TARAY(I))
         Y = RE*COS(AINRAY(I))*SIN(APRAY(I))*COS(TARAY(I)) +
     +         RE*COS(AINRAY(I))*COS(APRAY(I))*SIN(TARAY(I))
         Z = RE*SIN(AINRAY(I))*SIN(APRAY(I))*COS(TARAY(I)) +
     +         RE*SIN(AINRAY(I))*COS(APRAY(I))*SIN(TARAY(I))
```

```
*         CALCULATE LATITUDE
          ELARAY(I) = (ASIN(Z/RE)) * (180/3.14159)

*         TRAP 'X' AND 'Y' FOR ARCTAN IN CALCULATING LONGITUDE
          IF((Y .LE. 10) .AND. (Y .GE. 0.0)) THEN
               Y = 10.
          ELSEIF ((Y .GE.-10).AND.(Y .LE. 0.0)) THEN
               Y = -10.
          ENDIF
          IF((X .LE. 10) .AND. (X .GE. 0.0)) THEN
               X = 10.
          ELSEIF ((X .GE.-10).AND.(X .LE. 0.0)) THEN
               X = -10.
          ENDIF

*         CALCULATE LONGITUDE
          ELORAY(I) = (ATAN2(Y,X) - (EROT*TIMRAY(I))) * (180/3.14159)

*         MODIFY LONGITUDES TO ( -180 TO 180)
 420      IF (ELORAY(I) .LT. -180) THEN
               ELORAY(I) = ELORAY(I) + 360
               GOTO 420
          ENDIF
          I = I + 1
          GOTO 410
       ENDIF

*      SET DISSPLA
       CALL TEK618
       CALL RESET(3HALL)
       CALL YAXANG (0.)
       CALL PHYSOR(1.0,6.0)
       CALL XNAME(' ',1)
       CALL YNAME(' ',1)
       CALL AREA2D(7.5,3.75)
       CALL HEADIN ('GROUND TRACKS',100,1.5,1)
       CALL SCMPLX
       CALL MAPGR(-180.,90.,180.,-90.,30.,90.)
       CALL GRID (1,1)
       CALL MAPFIL ('MAPDTA')
       I = 1

*      IGNORE Boundary POINTS
 430   IF ((ELORAY(I) .LT. -175) .OR.
     +     (ELORAY(I) .GT. 175) .OR.
     +     (ELARAY(I) .LT. -85) .OR.
     +     (ELARAY(I) .GT. 85)) THEN
          I = I + 1
          GOTO 430
       ENDIF

       ITEMP = 1

*      LOAD FIRST POINT OF NEW PLOT SEGMENT
       IF (I .LE. NUM) THEN
```

```
         TLONG(ITEMP) = ELORAY(I)
         TLAT(ITEMP) = ELARAY(I)
         I = I + 1
*        IF ( I .GE. NUM) THEN
*            CALL POLY3
*            CALL CURVE(TLONG,TLAT,ITEMP,1)
*        ENDIF
      ENDIF

*     LOAD SECOND POINT IN LINE SEGMENT
      IF (I .LE. NUM) THEN
         ITEMP = ITEMP + 1
         TLONG(ITEMP) = ELORAY(I)
         TLAT(ITEMP) = ELARAY(I)
         I = I + 1
         IF ( I .GE. NUM) THEN
             CALL POLY3
             CALL NOCHEK
             CALL CURVE(TLONG,TLAT,ITEMP,1)
         ENDIF
      ENDIF

*     LOOP UNTIL SEGMENT REACHES EDGE OR NO MORE POINTS
 440  IF (I .LE. NUM) THEN

*        BOTH LAT AND LONG INCREASING
         IF((ELORAY(I - 2) .LE. ELORAY(I - 1)) .AND.
     +      (ELARAY(I - 2) .LE. ELARAY(I - 1))) THEN
             IF((ELORAY(I) .LT. -170) .OR.
     +          (ELARAY(I) .LT. -80)) THEN
                 CALL POLY3
                 CALL NOCHEK
                 CALL CURVE(TLONG,TLAT,ITEMP,1)
                 GOTO 430
             ELSE
                 ITEMP = ITEMP + 1
                 TLONG(ITEMP) = ELORAY(I)
                 TLAT(ITEMP) = ELARAY(I)
             ENDIF

*        BOTH LAT AND LONG DECREASING
         ELSEIF((ELORAY(I - 2) .GT. ELORAY(I - 1)) .AND.
     +          (ELARAY(I - 2) .GT. ELARAY(I - 1))) THEN
             IF((ELORAY(I) .GT.  170) .OR.
     +          (ELARAY(I) .GT.  80)) THEN
                 CALL POLY3
                 CALL NOCHEK
                 CALL CURVE(TLONG,TLAT,ITEMP,1)
                 GOTO 430
             ELSE
                 ITEMP = ITEMP + 1
                 TLONG(ITEMP) = ELORAY(I)
                 TLAT(ITEMP) = ELARAY(I)
             ENDIF

*        LAT INCREASING, LONG. DECREASING
```

```
      ELSEIF((ELORAY(I - 2) .GT. ELORAY(I - 1)) .AND.
     +       (ELARAY(I - 2) .LE. ELARAY(I - 1))) THEN
          IF((ELORAY(I) .GT.  170) .OR.
     +       (ELARAY(I) .LT. -80)) THEN
              CALL POLY3
              CALL NOCHEK
              CALL CURVE(TLONG,TLAT,ITEMP,1)
              GOTO 430
          ELSE
              ITEMP = ITEMP + 1
              TLONG(ITEMP) = ELORAY(I)
              TLAT(ITEMP) = ELARAY(I)
          ENDIF

*         LAT. DECREASING, LONG. INCREASING
      ELSEIF((ELORAY(I - 2) .LE. ELORAY(I - 1)) .AND.
     +       (ELARAY(I - 2) .GT. ELARAY(I - 1))) THEN
          IF((ELORAY(I) .LT. -170) .OR.
     +       (ELARAY(I) .GT.  80)) THEN
              CALL POLY3
              CALL NOCHEK
              CALL CURVE(TLONG,TLAT,ITEMP,1)
              GOTO 430
          ELSE
              ITEMP = ITEMP + 1
              TLONG(ITEMP) = ELORAY(I)
              TLAT(ITEMP) = ELARAY(I)
          ENDIF
      ENDIF
      IF( I .EQ. NUM) THEN
          CALL POLY3
          CALL NOCHEK
          CALL CURVE(TLONG,TLAT,ITEMP,1)
      ENDIF
      I = I + 1
      GOTO 440
   ENDIF


   CALL POLY3
   CALL NOCHEK
   CALL CURVE(TLONG,TLAT,ITEMP,1)


   CALL COMPLX
   CALL HEIGHT(.2)
   CALL THKFRM (0.03)
   CALL FRAME
   CALL RESET('COMPLX')
   CALL RESET('HEIGHT')
   CALL ENDGR (0)
   RETURN
   END

*******************************************************************************
```

```
      SUBROUTINE DATA(I,A,E,TFEA,TFSU,TFMO,TFDRA,PER,PI,TDI,TDA,TDE,
     +                 TDMM,TDMA,TDLAN,TDH,TDAP,MM,MA,LAN,H,AP,V,R)
*     THIS SUBROUTINE Displays THE ORBITAL DATA FOR BOTH THE PERIFOCAL
*     AND THE GROUND TRACK PLOTS.
*     REFER TO DISSPLA USER"S MANUAL FOR EXPLANATION OF DISSPLA
*     SUBROUTINES

      REAL I,MM,MA,LAN

      MU = 3.986012E+05

*     CALCULATE THE AVERAGE FORCES FROM THE TOTAL MAGNITUDE OF
*     FORCE CHANGES
      AVGFE = TFEA/50.0
      AVGFS = TFSU / 50.0
      AVGFM = TFMO / 50.0
      AVGFD = TFDRA / 50.0

*     CALCULATE ORBITAL ELEMENTS IN Usable UNITS
      PERH = PER/3600

      DI = I * (180.0/PI)
      DLAN = LAN * (180.0/PI)
      DAP = AP * (180.0/PI)

*     CALCULATE Average CHANGE IN ELEMENTS FOR ONE PERIOD
      AVGDI = TDI / 50.0
      AVGDA = TDA / 50.0
      AVGDE = TDE / 50.0
      AVGDMM = TDMM / 50.0
      AVGDMA = TDMA / 50.0
      AVGLAN = TDLAN / 50.0
      AVGDH = TDH / 50.0
      AVGDAP = TDAP / 50.0

*     CALCULATE RADIUS'S AND VELOCITIES
      ENR = ((V**2)/2) - (MU/R)
      RP = A*(1 - E)
      RA = A*(1 + E)
      VP = SQRT(2*(ENR + (MU/RP)))
      VA = SQRT(2*(ENR + (MU/RA)))


*     SET DISSPLA
      CALL RESET(3HALL)
      CALL SCMPLX
      CALL PHYSOR(0.0,0.0)
      CALL AREA2D(8.5,4.0)

*     PRINT DATA
      CALL MESSAG('I = $',100,0.25,3.67)
      CALL REALNO(DI,3,'ABUT','ABUT')
      CALL MESSAG(' DEG.$',100,'ABUT','ABUT')
      CALL MESSAG('   A = $',100,'ABUT','ABUT')
      CALL REALNO(A,1,'ABUT','ABUT')
      CALL MESSAG(' KM$',100,'ABUT','ABUT')
```

```
      CALL MESSAG(' E = $',100,'ABUT','ABUT')
      CALL REALNO(E,3,'ABUT','ABUT')
      CALL MESSAG('  PER = $',100,'ABUT','ABUT')
      CALL REALNO(PERH,2,'ABUT','ABUT')
      CALL MESSAG(' HOURS$',100,'ABUT','ABUT')

      CALL MESSAG('AVERAGE RATE OF CHANGE OF ELEMENTS PER SECOND $',
     +          100,1.0,3.0)

      CALL MESSAG('DI/DT = $',100,0.25,2.67)
      CALL REALNO(AVGDI,-2,'ABUT','ABUT')
      CALL MESSAG('  DA/DT = $',100,'ABUT','ABUT')
      CALL REALNO(AVGDA,-2,'ABUT','ABUT')
      CALL MESSAG('  DE/DT = $',100,'ABUT','ABUT')
      CALL REALNO(AVGDE,-2,'ABUT','ABUT')

      CALL MESSAG('DMM/DT = $',100,0.25,2.33)
      CALL REALNO(AVGDMM,-2,'ABUT','ABUT')
      CALL MESSAG('  DMA/DT = $',100,'ABUT','ABUT')
      CALL REALNO(AVGDMA,-2,'ABUT','ABUT')
      CALL MESSAG('  DLAN/DT = $',100,'ABUT','ABUT')
      CALL REALNO(AVGLAN,-2,'ABUT','ABUT')

      CALL MESSAG('DH/DT = $',100,0.25,2.00)
      CALL REALNO(AVGDH,-2,'ABUT','ABUT')
      CALL MESSAG('  DAP/DT = $',100,'ABUT','ABUT')
      CALL REALNO(AVGDMA,-2,'ABUT','ABUT')

      CALL MESSAG('AVERAGE MAGNITUDE OF FORCES PER UNIT MASS (KM/S**2)
     +$',100,1.0,1.67)

      CALL MESSAG('EARTH = $',100,0.10,1.33)
      CALL REALNO(AVGFE,-1,'ABUT','ABUT')
      CALL MESSAG(' MOON = $',100,'ABUT','ABUT')
      CALL REALNO(AVGFM,-1,'ABUT','ABUT')
      CALL MESSAG(' SUN = $',100,'ABUT','ABUT')
      CALL REALNO(AVGFS,-1,'ABUT','ABUT')
      CALL MESSAG(' DRAG = $',100,'ABUT','ABUT')
      CALL REALNO(AVGFD,-1,'ABUT','ABUT')

      CALL MESSAG('PERIGEE$',100,2.75,1.0)
      CALL MESSAG('                    Apogee$',100,'ABUT','ABUT')

      CALL MESSAG('RADIUS (KM)$',100,0.25,0.67)
      CALL MESSAG('RP =$',100,2.75,0.67)
      CALL REALNO(RP,1,'ABUT','ABUT')
      CALL MESSAG('      $',100,'ABUT','ABUT')
      CALL MESSAG('  RA =$',100,'ABUT','ABUT')
      CALL REALNO(RA,1,'ABUT','ABUT')

      CALL MESSAG('VELOCITY (KM/SEC)$' 100,0.25,0.33)
      CALL MESSAG('VP =$',100,2.75,0.33)
      CALL REALNO(VP,2,'ABUT','ABUT')
      CALL MESSAG('       $',100,'ABUT','ABUT')
      CALL MESSAG('  VA =$',100,'ABUT','ABUT')
      CALL REALNO(VA,2,'ABUT','ABUT')
```

```
      CALL RESET('COMPLX')
      CALL ENDGR(0)
      RETURN
      END
```

# APPENDIX B. COORDINATE SYSTEMS

## A. 'IJK' : GEOCENTRIC - EQUATORIAL

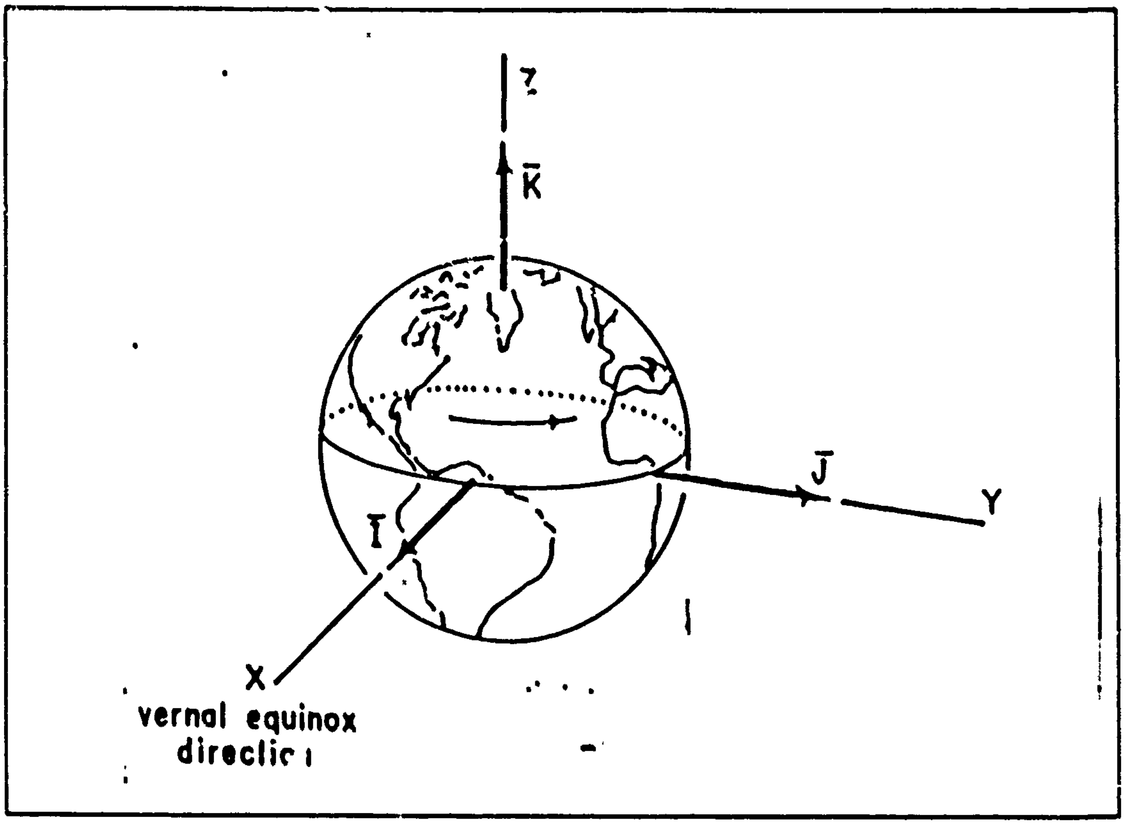



**Figure 3. Geocentric-equatorial coordinate system**

The geocentric-equatorial system as seen in Figure 3 has its origin at the earth's center. The fundamental plane is in the equator and the positive X-axis points in the vernal equinox direction. The Z-axis points in the direction of the north pole. This system is not fixed to the earth and turning with it; rather, the geocentric-equatorial frame is nonrotating with respect to the stars (except for precession of the equinoxes) and the earth turns relative to it. Unit vectors, $\bar{I}$, $\bar{J}$, and $\bar{K}$ shown in Figure 3, lie along the X, Y, and Z respectively. [Ref. 1: p.55]

## B. 'PQW': PERIFOCAL

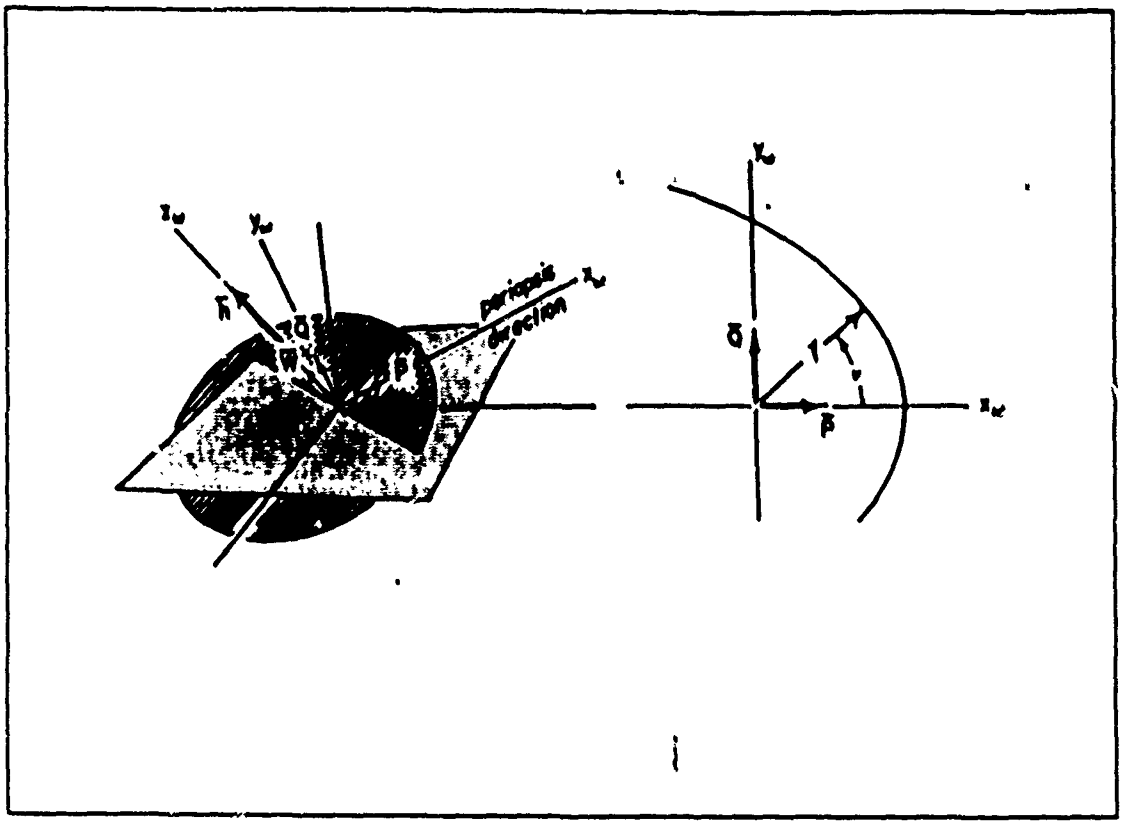



Figure 4. Perifocal coordinate system

The perifocal coordinate system has its fundamental plane in the plane of the satellite's orbit as seen in Figure 4. The coordinate axis are named, $X_\omega$, $Y_\omega$ and $Z_\omega$ .. The $X_\omega$ axis points toward the perigee; the $Y_\omega$ axis is rotated 90 degrees in the direction of orbital motion and lies in the orbital plane; the $Z_\omega$ axis along $\bar{h}$ completes the right-handed perifocal system. Unit vectors in the direction of $X_\omega$, $Y_\omega$ and $Z_\omega$ are called $\bar{P}$, $\bar{Q}$ and $\bar{W}$ respectively. [Ref. 1: p.57]

## C. 'RSW' : ORBITAL

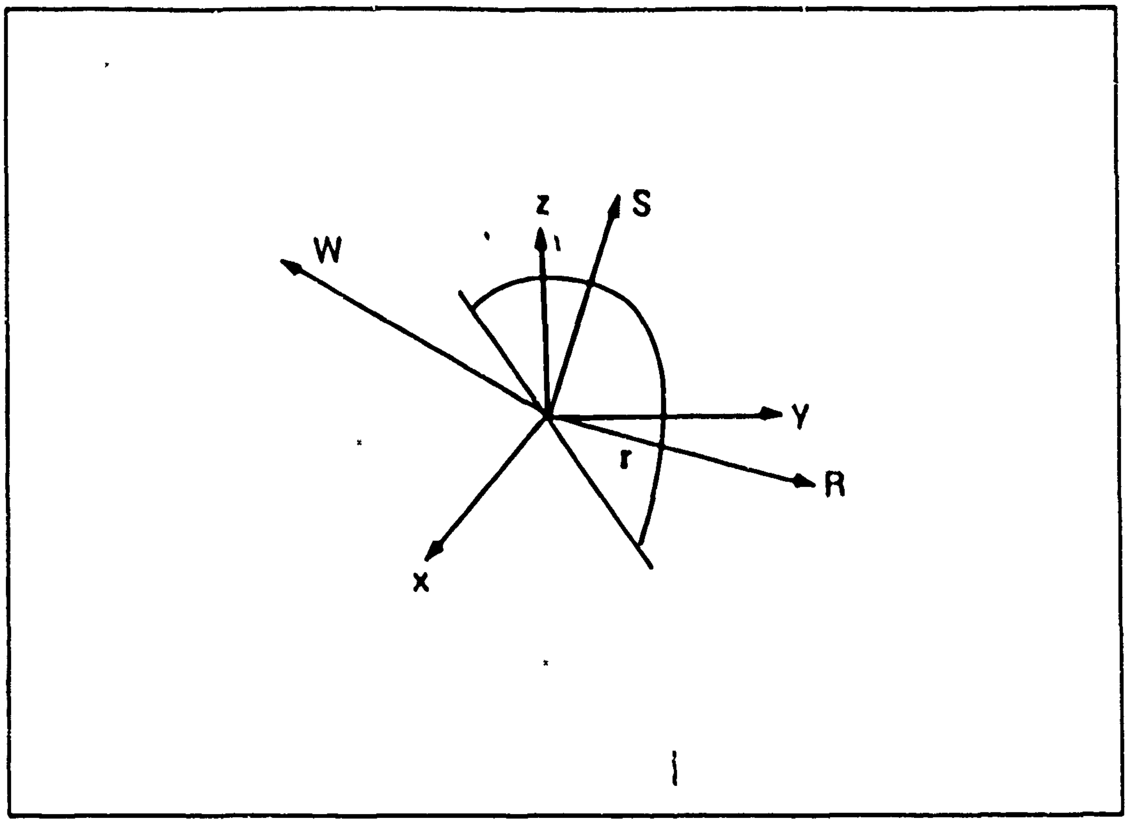



**Figure 5. Orbital coordinate system**

(Figure 9.4-1, Ref. 1)

The orbital coordinate system has its principle axis, R (unit vector r), along the instantaneous radius vector, r as seen in Figure 5. The axis S is rotated 90 degrees from R in the direction of increasing true anomaly. The third axis, W, is perpendicular to both R and S. Note that this coordinate system is simply rotated $\nu_0$ from the PQW perifocal system. [Ref. 1: p.398]

## D. COORDINATE TRANSFORMATIONS

The coordinate transformations, for the previous coordinate systems, use angular rotations about the axis to evaluate the transformation matrix. The matrix elements $r_{ij}$ are calculated, then applied to the old vector to get the vector in the new coordinate system. The following orbital elements are used:

$\Omega$ = longitude of ascending node

$\omega$ = argument of perigee

i = inclination

$u_0$ = argument of latitude

$v_0$ = true anomaly

The coordinate transformations follow [Ref. 1: p.74-83]

1. PQW to IJK

$$r_{11} = \cos\Omega\cos\omega - \sin\Omega\sin\omega\cos i$$
$$r_{12} = -\cos\Omega\sin\omega - \sin\Omega\cos\omega\cos i$$
$$r_{13} = \sin\Omega\cos\omega$$
$$r_{21} = \sin\Omega\cos\omega + \cos\Omega\sin\omega\cos i$$
$$r_{22} = -\sin\Omega\sin\omega + \cos\Omega\cos\omega\cos i$$
$$r_{23} = -\cos\Omega\sin i$$
$$r_{31} = \sin\omega\sin i$$
$$r_{32} = \cos\omega\sin i$$
$$r_{33} = \cos i$$
$$\vec{I} = r_{11}\vec{P} + r_{12}\vec{Q} + r_{13}\vec{W}$$
$$\vec{J} = r_{21}\vec{P} + r_{22}\vec{Q} + r_{23}\vec{W}$$
$$\vec{K} = r_{31}\vec{P} + r_{32}\vec{Q} + r_{33}\vec{W}$$

2. IJK to PQW (inverse of #1)

$$\vec{P} = r_{11}\vec{I} + r_{21}\vec{J} + r_{31}\vec{K}$$
$$\vec{Q} = r_{12}\vec{I} + r_{22}\vec{J} + r_{32}\vec{K}$$
$$\vec{W} = r_{13}\vec{I} + r_{23}\vec{J} + r_{33}\vec{K}$$

3. IJK to RSW

$$r_{11} = \cos\Omega\cos u_0 - \sin\Omega\sin u_0\cos i$$
$$r_{12} = \sin\Omega\cos u_0 + \sin u_0\cos\Omega\cos i$$
$$r_{13} = \sin i\sin u_0$$
$$r_{21} = -\cos\Omega\sin u_0 - \sin\Omega\cos u_0\cos i$$
$$r_{22} = -\sin\Omega\sin u_0 + \cos\Omega\cos u_0\cos i$$
$$r_{23} = \cos u_0\sin i$$
$$r_{31} = \sin\Omega\sin i$$
$$r_{32} = -\cos\Omega\sin i$$
$$r_{33} = \cos i$$
$$\vec{R} = r_{11}\vec{I} + r_{12}\vec{J} + r_{13}\vec{K}$$
$$\vec{S} = r_{21}\vec{I} + r_{22}\vec{J} + r_{23}\vec{K}$$
$$\vec{W} = r_{31}\vec{I} + r_{32}\vec{J} + r_{33}\vec{K}$$

4. RSW to IJK (inverse of #3)

$$\vec{I} = r_{11}\vec{R} + r_{21}\vec{S} + r_{31}\vec{W}$$
$$\vec{J} = r_{12}\vec{R} + r_{22}\vec{S} + r_{32}\vec{W}$$
$$\vec{K} = r_{13}\vec{R} + r_{23}\vec{S} + r_{33}\vec{W}$$

5. PQW to RSW

$r_{11} = \cos v_0$
$r_{12} = \sin v_0$
$r_{13} = 0.0$
$r_{21} = -\sin v_0$
$r_{22} = \cos v_0$
$r_{23} = 0.0$
$r_{31} = 0.0$
$r_{32} = 0.0$
$r_{33} = 1.0$

$$\vec{R} = r_{11}\vec{P} + r_{12}\vec{Q} + r_{13}\vec{W}$$
$$\vec{S} = r_{21}\vec{P} + r_{22}\vec{Q} + r_{23}\vec{W}$$
$$\vec{W} = r_{31}\vec{P} + r_{32}\vec{Q} + r_{33}\vec{W}$$

6. RSW to PQW (inverse of #5)

$$\vec{P} = r_{11}\vec{R} + r_{21}\vec{S} + r_{31}\vec{W}$$
$$\vec{Q} = r_{12}\vec{R} + r_{22}\vec{S} + r_{32}\vec{W}$$
$$\vec{W} = r_{13}\vec{R} + r_{23}\vec{S} + r_{33}\vec{W}$$

# APPENDIX C. ORBITAL ELEMENTS

The user is assumed to be studying orbital mechanics and should understand the orbital elements and how to calculate them. A brief description of the elements and the equations used to calculate the elements follow. For a detailed explanation of the elements and the equations to calculate them refer to Chapters 1 and 2 of reference 1. Figure 6 on page 83 shows the orbital elements in the Geocentric-Equatorial and perifocal coordinate system.

1. Angular Momentum (h):
   The specific angular momentum is a constant of the motion of the satellite, defined as $\vec{h} = \vec{r} x \vec{v}$.

$$\vec{h} = \vec{r} x \vec{v} = h_I \vec{I} + h_J \vec{J} + h_K \vec{K}$$

$$h_I = r_J v_K - r_K v_J$$

$$h_J = r_K v_I - r_I v_K$$

$$h_K = r_I v_J - r_J v_I$$

$$h = \sqrt{h_I^2 + h_J^2 + h_K^2}$$

2. Node Vector (n):
   The node vector is a vector pointing along the line of nodes in the direction of the ascending node.

$$\vec{n} = \vec{K} x \vec{h} = -h_J \vec{I} + h_I \vec{J}$$

$$n = \sqrt{h_J^2 + h_I^2}$$

3. Semi-latus rectum (p):
   The semi-latus rectum is a geometric constant of the conic section.

$$p = \frac{h^2}{\mu}$$

4. Eccentricity (e):
   The eccentricity is a constant defining the shape of the conic orbit.

$$\vec{e} = \frac{1}{\mu} [(v^2 - \frac{\mu}{r})\vec{r} - (\vec{r}\,\vec{v})\vec{v}]$$

$$e = |\vec{e}|$$

5. Semi-major axis (a):

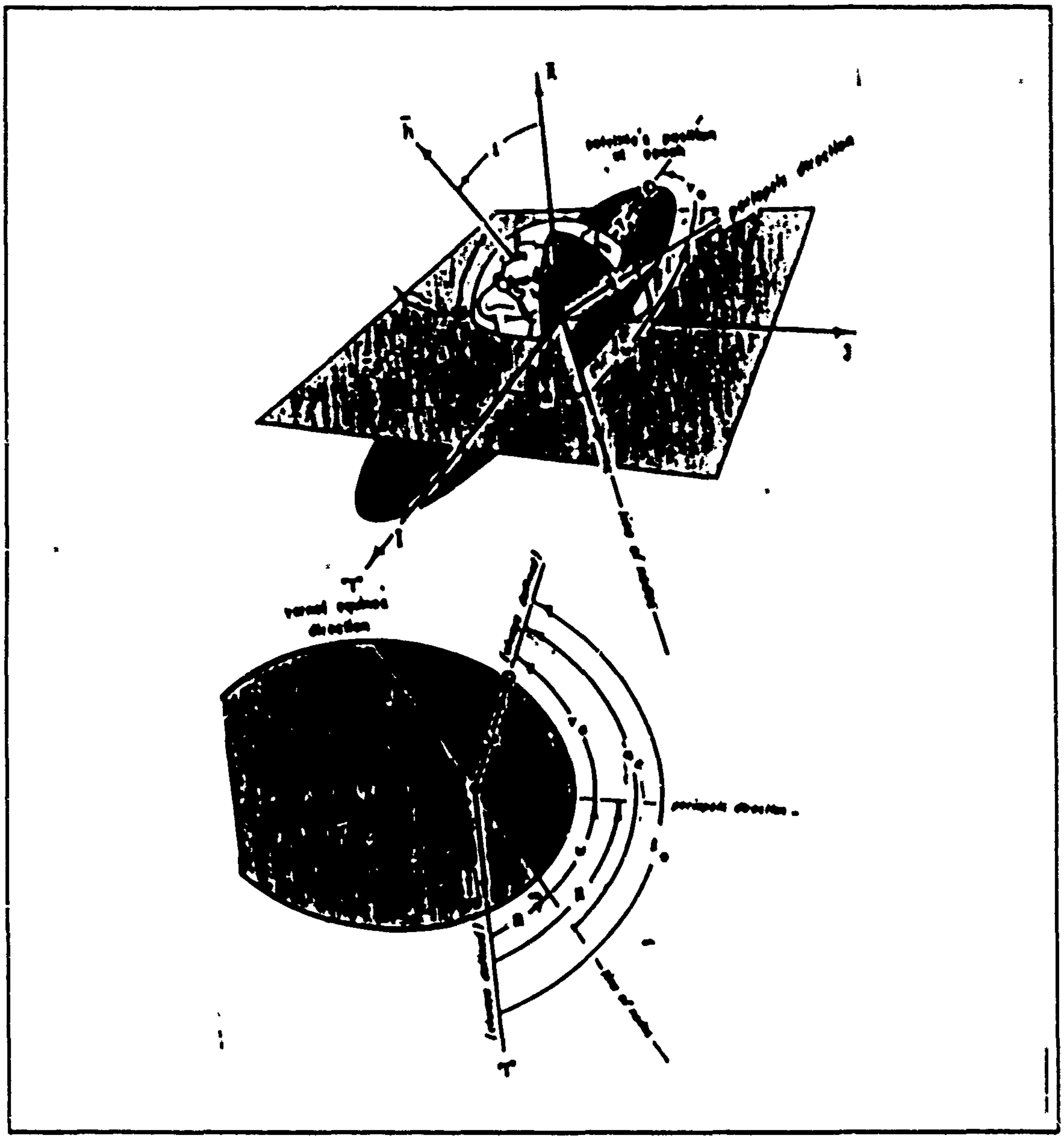


**Figure 6. Orbital elements**

The semi-major axis is a constant defining the size of the orbit.

$$a = \frac{(1 - e^2)}{p}$$

6. Inclination (i):

The inclination is the angle between the 'K' unit vector in the 'IJK' system and the angular momentum vector, 'h'.

$$i = \cos^{-1}\left(\frac{\bar{h}\,\bar{K}}{h}\right) = \cos^{-1}\left(\frac{h_k}{h}\right)$$

7. Longitude of ascending node ($\Omega$):

The longitude of the ascending node is the angle in the fundamental plane, between the 'I' unit vector and the point where the satellite crosses through the fundamental plane in a northerly direction (ascending node) measured counter-clockwise when viewed from the north side of the fundamental plane.

$$\Omega = \cos^{-1}\left(\frac{n_i}{n}\right)$$

8. Argument of perigee ($\omega$):

The argument of perigee is the angle in the plane of the satellite's orbit, between the ascending node and the perigee point, measured in the direction of the satellite's motion.

$$\omega = \cos^{-1}\left(\frac{\bar{n}\,\bar{e}}{ne}\right) = \cos^{-1}\frac{(n_i e_i + n_j e_j)}{ne}$$

9. True anomaly at epoch ($v_0$):

The true anomaly at epoch is the angle in the plane of the satellite's orbit, between perigee and the position of the satellite at a particular time, $t_0$, called the "epoch".

$$v_0 = \cos^{-1}\left(\frac{\bar{e}\,\bar{r}}{er}\right)$$

10. Argument of latitude ($u_0$):

The argument of latitude is the angle in the plane of the orbit, between the ascending node and the radius vector to the satellite at time $t_0$.

$$u_0 = \cos^{-1}\left(\frac{\bar{n}\,\bar{r}}{nr}\right)$$

11. Longitude of perigee ($\Pi$):

The longitude of perigee is the angle from 'I' to perigee measured eastward to the ascending node and then in the orbital plane to perigee.

$$\Pi = \Omega + \omega$$

12. True longitude at epoch ($l_0$):

The true longitude at epoch is the angle between 'I' and $r_0$ (the radius vector to the satellite at $t_0$ measured eastward to the ascending node and then in the orbital plane to $r_0$.

$$l_0 = \omega + \Omega + v_0$$

13. Period (per):

The period is the time the for the satellite to complete one orbit.

$$Per = 2\sqrt{\frac{a^3}{\mu}}$$

14. Eccentric anomaly (EA):

The eccentric anomaly is the angle between the perigee and a position on an auxiliary circle circumscribed about the ellipse where a perpendicular line to the major axis has been extended from the epoch location of the satellite to the auxiliary circle.

$$EA = \cos^{-1}\frac{e + \cos(v)}{1 + e\cos(v)}$$

15. Mean motion (n'):

The mean motion is defined below:

$$n' = \sqrt{\frac{\mu}{a^3}}$$

16. Mean anomaly (MA):

The mean anomaly is defined below:

$$MA = n'(t - T) = EA - e\sin(EA)$$

17. Time of flight (TF):

The time of flight is the elapsed time from when the satellite was at perigee to the current epoch.

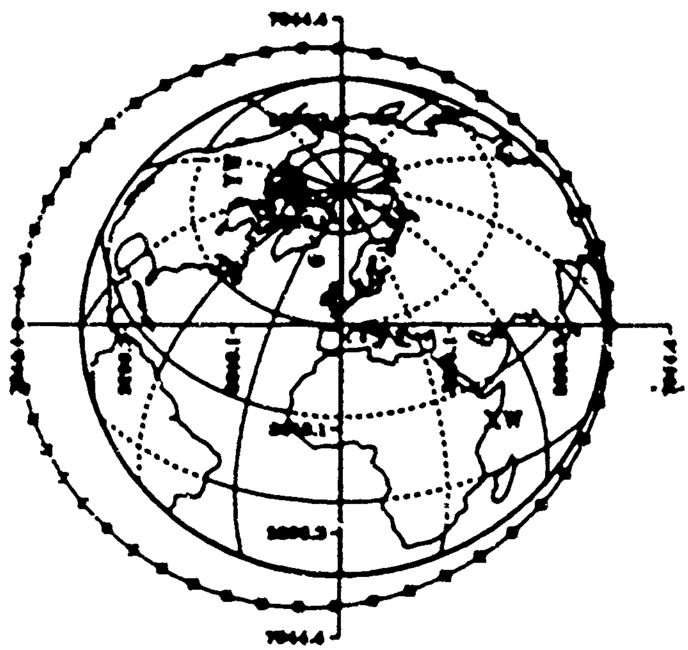
$$(t - T) = \sqrt{\frac{a^3}{\mu}}\,(EA - e\sin(EA))$$

# APPENDIX D. SAMPLE ORBITS

To demonstrate the capabilities of the program, a variety of orbital plots will follow:

1. Low earth orbit (LEO).

**PERIFOCAL COORDINATE SYSTEM**




I = 45.000 DEG. A = 7222.2 KM E = 0.100 PER = 1.70 HOURS

AVERAGE RATE OF CHANGE OF ELEMENTS PER SECOND

DI/DT = 0.00 DA/DT = 0.00 DE/DT = 0.00

DMM/DT = 0.00 DMA/DT = 0.00 DLAN/DT = 0.00

DH/DT = 0.00 DAP/DT = 0.00

AVERAGE MAGNITUDE OF FORCES PER UNIT MASS (KM/S**2)

EARTH = 0.0 MOON = 0.0 SUN = 0.0 DRAG = 0.0

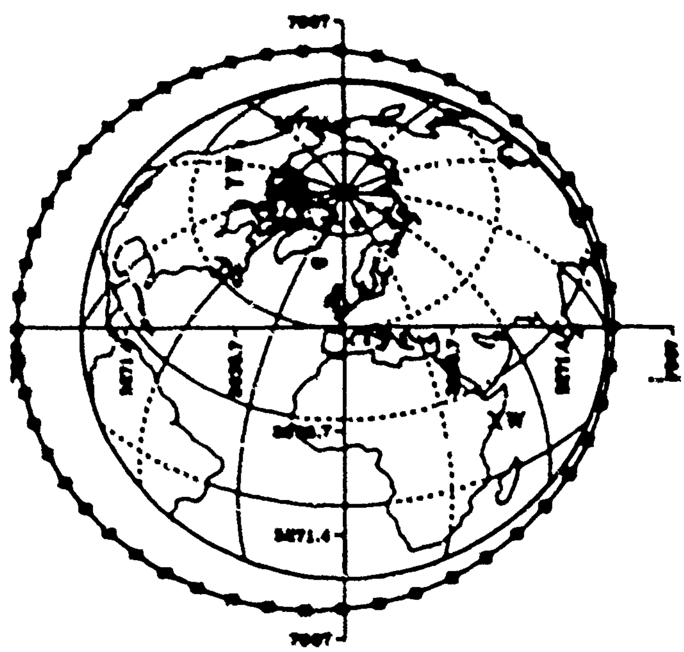
| | PERIGEE | Apogee |
|---|---|---|
| RADIUS (KM) | RP = 6500.0 | RA = 7944.4 |
| VELOCITY (KM/SEC) | VP = 8.21 | VA = 6.72 |

Figure 7. Unperturbed Low Earth Orbit (LEO)

Figure 7 shows the perifocal plot of a satellite in an unperturbed low earth orbit (LEO). The initial parameters of the orbit were entered as follows:

radius of perigee (RP) = 6500 km
eccentricity (e) = 0.1
inclination (i) = 45 degrees.

PERIFOCAL COORDINATE SYSTEM



I = 44.998 DEG. A = 7203.3 KM E = 0.098 PER = 1.69 HOURS

AVERAGE RATE OF CHANGE OF ELEMENTS PER SECOND

DI/DT = 4.20*10^[illegible] DA/DT = 8.36*10^[illegible] DE/DT = 1.30*10^[illegible]

DMM/DT = 1.80*10^[illegible] DMA/DT = 8.21*10^[illegible] DLAN/DT = 9.48*10^[illegible]

DH/DT = 5.83*10^[illegible] DAP/DT = 9.21*10^[illegible]

AVERAGE MAGNITUDE OF FORCES PER UNIT MASS (KM/S**2)

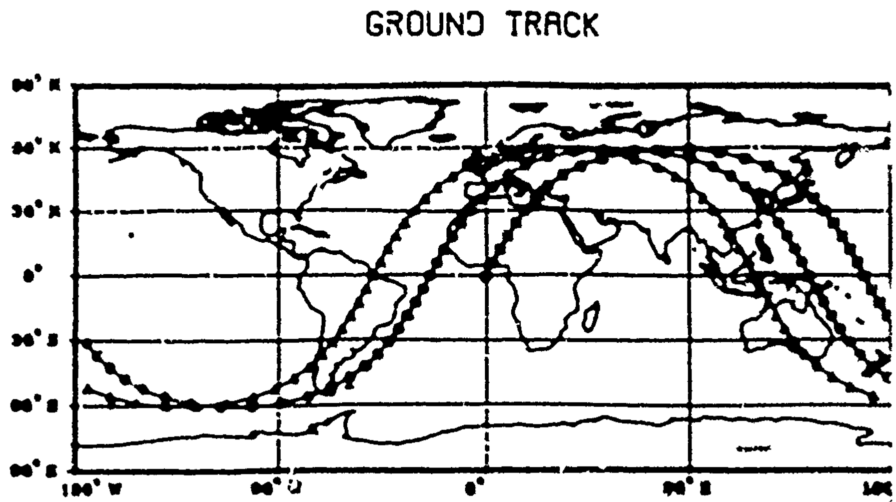
EARTH = 9.8*10^[illegible] MOON = 9.4*10^[illegible] SUN = 4.3*10^[illegible] DRAG = 1.4*10^[illegible]

| | PERIGEE | Apogee |
|---|---|---|
| RADIUS (KM) | RP = 6499.6 | RA = 7907.0 |
| VELOCITY (KM/SEC) | VP = 8.20 | VA = 6.74 |

Figure 8. Perturbed Low Earth Orbit (LEO)

With perturbing forces applied to the previous LEO, the drag force will be the dominate perturbing force. The drag will act as a negative velocity change applied in the area of perigee, with the result of decreasing the semi-major axis length, this in effect will decrease the eccentricity of the orbit, as can be seen by comparing the orbital data of the unperturbed LEO in Figure 7 on page 86 with the orbital data of the perturbed LEO in Figure 8.

2. Circular orbit.




I = 59.967 DEG. A = 6986.1 KM E = 0.000 PER = 1.61 HOURS

AVERAGE RATE OF CHANGE OF ELEMENTS PER SECOND

DI/DT = $4.02*10^{-7}$ DA/DT = $9.74*10^{-8}$ DE/DT = $5.93*10^{-7}$

DMM/DT = $2.26*10^{-6}$ DMA/DT = $1.52*10^{-6}$ DLAN/DT = $7.32*10^{-7}$

DH/DT = $5.86*10^{-6}$ DAP/DT = $1.52*10^{-6}$

AVERAGE MAGNITUDE OF FORCES PER UNIT MASS (KM/S**2)

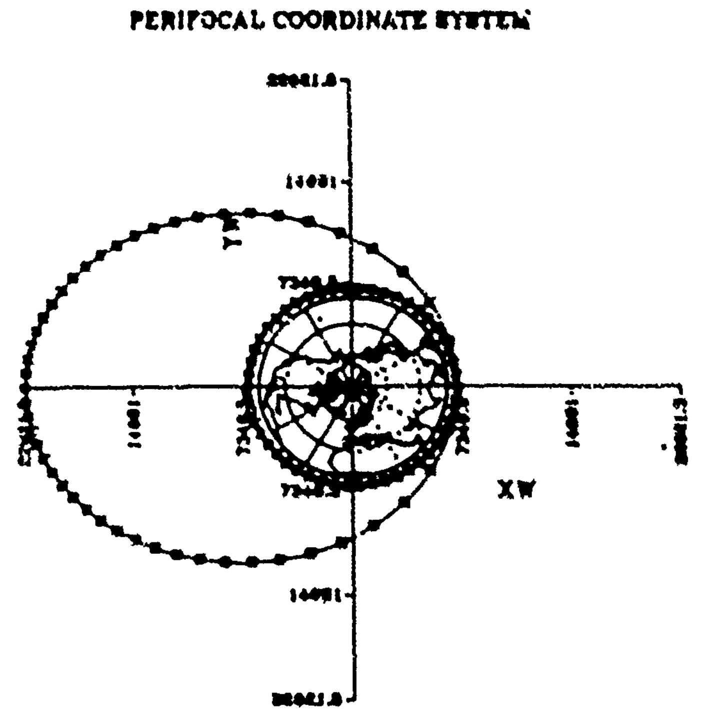
EARTH = $1.2*10^{-3}$ MOON = $8.8*10^{-10}$ SUN = $4.3*10^{-10}$ DRAG = $3.0*10^{-10}$

| | PERIGEE | Apogee |
|---|---|---|
| RADIUS (KM) | RP = 6984.4 | RA = 6987.8 |
| VELOCITY (KM/SEC) | VP = 7.56 | VA = 7.55 |

Figure 9. Circular Orbit

An example of the plot of the ground track of a sequence of three 60 degree inclined perturbed circular orbits with a radius of 7000 km is shown in Figure 9. The sequence of orbits displays the precession of the orbit around the earth.

3. Transfer orbit.




i = 0.000 DEG. A = 14510.8 KM E = 0.518 PER = 4.83 HOURS

AVERAGE RATE OF CHANGE OF ELEMENTS PER SECOND

DI/DT = 0.00 DA/DT = 0.00 DE/DT = 0.00

DMM/DT = 0.00 DMA/DT = 0.00 DLAN/DT = 0.00

DH/DT = 0.00 DAP/DT = 0.00

AVERAGE MAGNITUDE OF FORCES PER UNIT MASS (KM/S**2)

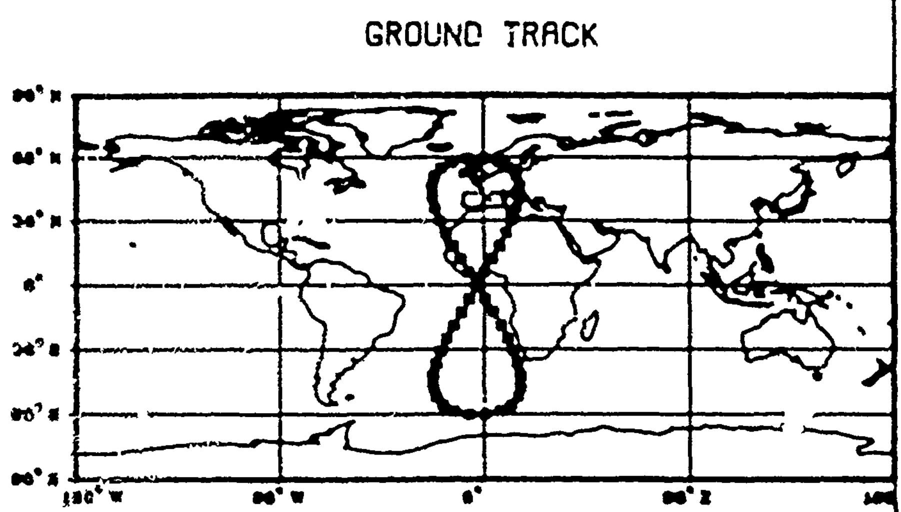
EARTH = 0.0 MOON = 0.0 SUN = 0.0 DRAG = 0.0

| | PERIGEE | Apogee |
|---|---|---|
| RADIUS (KM) | RP = 7000.0 | RA = 22021.5 |
| VELOCITY (KM/SEC) | VP = 9.30 | VA = 2.95 |

Figure 10. Transfer Orbit

The transfer orbit between a circular, equatorial LEO and a molniya orbit (high eccentric orbit) is shown in Figure 10. A velocity increase of 1.75 km/s was applied at the perigee to simulate a perigee kick to boost the satellite into the molniya orbit. A similar velocity change could then be applied at apogee to create a high altitude circular orbit, or a negative velocity change applied at perigee could be used to bring the satellite back to a LEO.

4. Geosynchronous orbit




I = 60.000 DEG. A = 42234.6 KM E = 0.000 PER = 23.99 HOURS

AVERAGE RATE OF CHANGE OF ELEMENTS PER SECOND

DI/DT = $1.04 \cdot 10^{-8}$ DA/DT = $1.25 \cdot 10^{-4}$ DE/DT = $1.08 \cdot 10^{-8}$

DMM/DT = $3.26 \cdot 10^{-10}$ DMA/DT = $3.02 \cdot 10^{-8}$ DLAN/DT = $1.89 \cdot 10^{-8}$

DH/DT = $3.55 \cdot 10^{-4}$ DAP/DT = $3.02 \cdot 10^{-8}$

AVERAGE MAGNITUDE OF FORCES PER UNIT MASS (KM/S**2)

EARTH = $9.3 \cdot 10^{-8}$ MOON = $4.9 \cdot 10^{-8}$ SUN = $2.6 \cdot 10^{-8}$ DRAG = $3.1 \cdot 10^{-10}$

| | PERIGEE | Apogee |
|---|---|---|
| RADIUS (KM) | RP = 42233.9 | RA = 42235.3 |
| VELOCITY (KM/SEC) | VP = 3.07 | VA = 3.07 |

Figure 11. Geosynchronous Orbit

The ground track of a perturbed geosynchronous orbit inclined 60 degrees is shown in Figure 11. The orbit displays the figure eight typical with inclined geosynchronous orbits.

## LIST OF REFERENCES


1. Bate, R.R., Mueller, D.D., and White, J.E., *Fundamentals of Astrodynamics*, Dover Publications, Inc., 1971.

2. Martin Marrieta Corporation, Space Systems Division, *Orbital Flight Handbook*, volume 1, 1963.

3. Agrawal, B.N., *Design of Geosynchronous Spacecraft*, Prentice-Hall, Inc., 1968.

4. Roy, A.E., *Orbital Motion*, Adam Hilger Ltd., 1982.

5. Computer Associates, *DISSPLA User's Manual*, version 10.0, 1987.

## INITIAL DISTRIBUTION LIST

| | | No. Copies |
|---|---|---|
| 1. | Defense Technical Information Center<br>Cameron Station<br>Alexandria, VA 22304-6145 | 2 |
| 2. | Library, Code 0142<br>Naval Postgraduate School<br>Monterey, CA 93943-5002 | 2 |
| 3. | Commander<br>Naval Space Command<br>Attn: Code N3<br>Dahlgren, VA 22448 | 1 |
| 4. | Commander<br>United States Space Command<br>Attn: Technical Library<br>Peterson AFB, CO 80914 | 1 |
| 5. | Dirrector<br>Navy Space System Division<br>Chief of Naval Operations (OP-943)<br>Washington, D.C. 20305-2000 | 1 |
| 6. | Superintendent<br>Naval Postgraduate School<br>Space System Academic Group<br>Attn: Prof. R. Panholzer (Code 62Pz)<br>Monterey, CA 93943-5000 | 1 |
| 7. | Superintendent<br>Naval Postgraduate School<br>Attn: Prof. E. Milne (Code 61Mn)<br>Monterey, CA 93943-5000 | 1 |
| 8. | Superintendent<br>Naval Postgraduate School<br>Attn: Prof. D. Danielson (Code 53Dd)<br>Monterey, CA 93943-5000 | 1 |
| 9. | Commanding Oficer<br>Attn: LT Anthony Vraa<br>Department Head School Class 111<br>Surface Warfare Officer School<br>Newport, RI 02841-5012 | 2 |